目　录

第一章 MATLAB 的基本知识及入门

本章主要包括MATLAB 简介、基本知识、基本命令的使用。学生通过学习，进入到 MATLAB 环境中操作演算，了解 MATLAB 的全貌，可以迅速产生兴趣，为后面的学习打下基础。

1.1 MATLAB 简介

1980 年美国的 Cleve Moler 博士在新墨西哥大学讲授线性代数课程时，发现采用高级语言编程极为不便，故开发了 MATLAB (Matrix Laboratory)，即矩阵实验室。它是进行科学计算的软件系统，经数年试用之后，于 1984 年推出了该软件的正式版本，其核心是采用 C 语言编写。MATLAB 环境使得矩阵运算变得非常容易，后来的版本又增添了图形生成、图像处理及多媒体功能，应用范围越来越广泛，逐渐活跃在工程领域中。不久，Moler 博士等一批数学家与软件专家组建了 MathWorks 的软件开发公司，并继续从事 MATLAB 的研究和开发。

为了正确合理地设计一个复杂控制系统，然后对它进行进一步的分析、仿真、校正，1990～1992 年 MathWorks 软件公司为 MATLAB 提供了新的控制系统模型图形输入与仿真工具 SIMULINK。此软件可利用鼠标器在窗口上设计出所需要的控制系统模型，然后利用该软件提供的功能来对系统直接进行仿真，观察分析其系统性能。显然，这种做法使得一个很复杂系统的建成变得相当容易。SIMULINK 的出现，更使得MATLAB 为控制系统的仿真与其在控制系统CAD 中的应用打开了崭新的局面。

目前的 MATLAB 已经成为国际上最为流行的软件之一，它除了传统的交互式编程之外，还提供了丰富可靠的矩阵运算、图形绘制、数据处理、图像处理、语言编程等便利工具，出现了以各种 MATLAB 为基础的实用工具箱，广泛地应用于自动控制、图像信号处理、模糊推理、神经网络、小波变换、信号分析、振动理论、时序分析与建模、化学统计学、优化设计等领域，并表现出一般高级语言难以比拟的优势。在欧美高等院校，MATLAB 已经成为应用线性代数、自动控制理论、数理统计、数字信号处理、时间序列分析、动态系统仿真等高级课程的基本教学工具，成为攻读学位的大学生、硕士生、博士生必须掌握的基本技能。在设计研究单位和工业部门，MATLAB 被广泛地用于研究和解决各种具体工程问题。较为常见的

MATLAB 工具箱主要包括：

(1) 控制系统工具箱(Control Systems Toolbox)；

(2) 系统辨识工具箱(System Identification Toolbox)；

(3) 鲁棒控制工具箱(Robust Control Toolbox)；

(4) 多变量频率设计工具箱(Multivariable Frequency DesignToolbox)；

(5) μ 分析与综合工具箱(μ-analysis And Synthesis Toolbox)；

(6) 神经网络工具箱(Neural Network Toolbox)；

(7) 最优化工具箱(Optimization Toolbox)；

(8) 小波分析工具箱(Wavelet Toolbox)；

(9) 通讯工具箱(Communication Toolbox)；

(10) 财政金融工具箱(Finanical Toolbox)；

(11) 频率域系统辨识工具箱(Frequency Domain System Identification Toolbox)；

(12) 模糊逻辑工具箱(Fuzzy Logic Toolbox)；

(13) 高阶谱分析工具箱(Higher Order Spectral Analysis Toolbox)；

(14) 图像处理工具箱(Image Processing Toolbox)；

(15) 线性矩阵不等式控制工具箱(LMI Control Toolbox)；

(16) 模型预测控制工具箱(Model Predictive Control Toolbox)；

(17) 偏微分方程工具箱(Partial Differential Equation Toolbox)；

(18) 信号处理工具箱(Signal Processing Toolbox)；

(19) 样条工具箱(Spline Toolbox)；

(20) 统计工具箱(Statistics Toolbox)；

(21) 符号数学工具箱(Symbolic Math Toolbox)；

(22) 电厂系统(Power System)。

1.2 MATLAB 的启动

如果你是初学者，可以利用菜单、快捷键或文件夹三种方式进入 MATLAB 工作窗口。但最基本、最容易的方法是通过菜单，双击 MATLAB 级联菜单上的图标。

MATLAB 启动后，自动运行 x:\matlab\toolbox\local\matlabrc.m 文件，首先按 pathdef.m 文件的要求设置系统路径，然后在工作窗最上方显示初始提示信息，设置系统环境，运行 startup.m 文件。若 MATLAB 是在英文 Win9x 平台上运行，那么 MATLAB 工作窗中的第三行将出现 MATLAB 环境提示符号“>>”和光标。在中文 Win9x 平台上的 MATLAB 工作窗中，将不显示提示符号“>>”，而

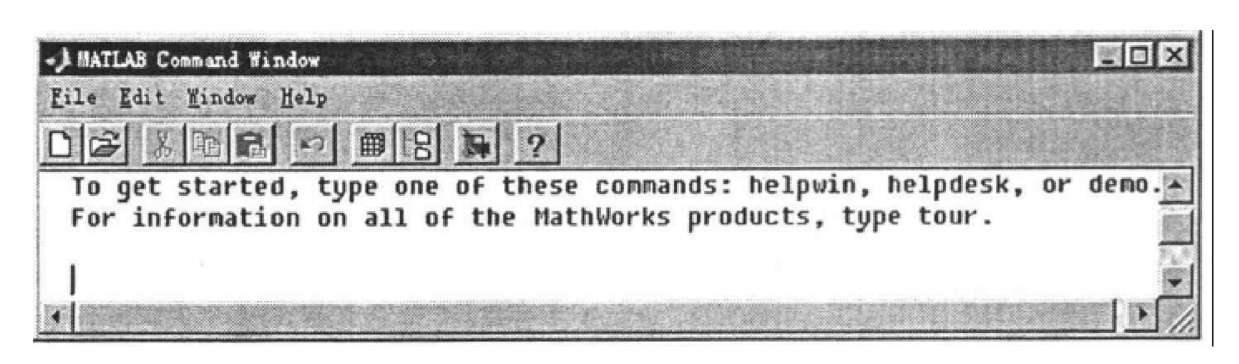


图1.2-1　MATLAB的工作窗

只有光标位置符，如图1.2-1所示。当MATLAB工作窗打开以后，你就可以在工作窗里进行各种运算操作。为此，本章将主要介绍命令行的基本操作。MATLAB中的一些常用操作命令见表1-1。

表1-1　**MATLAB命令窗中的部分常用命令**

demo	演示程序	type	显示文件内容
help	在线帮助指令	what	列出当前目录下的m文件、mat文件和mex文件
lookfor	关键词检索	which	确定指定函数和文件的位置
path	设置MATLAB的搜索路径	who	列出工作内存中的变量名
pathtool	搜索路径管理器	whos	列出工作内存中的变量细节
clear	从内存中清除变量和函数	pack	合并工作内存中的碎块
exist	检查变量或文件的存在性	size	确定矩阵的维数
length	确定向量的长度	workspace	工作内存浏览器
quit	关闭和退出MATLAB	dir	列出指定目录下的文件和子目录清单
clc	擦除工作窗中的显示内容	cd	改变当前工作子目录
clf	擦除当前图形窗中的图形	disp	显示字符串内容
hold	控制当前图形对象是否被刷新	echo	控制运行文件指令是否显示的开关
load	从磁盘文件中调入数据变量	save	把内存数据变量存入磁盘文件中

1.3　变量、表达式和语句

与其他计算机语言相似，变量可由字母、数字和下划线三种字符组成，且第一

个字符必须为字母，但有大小写之分。表达式是由算术运算符、关系运算符和逻辑运算符将常量或变量连接的式子。例如：

```
a=3;
b=2;
c=a>b
c=
   1
```

这里“c=”表示显示的答案，以后不再赘述。

语句可以是由表达式、函数或是变量、赋值号、表达式的组合，形成计算机最基本的执行单元。

例如：

```
1233/34;
sqrt(3);
a=b+c;
```

MATLAB还有一些特殊变量，见表1-2。

表1-2 **MATLAB的特殊变量**

变量名	说明
ans	每一次运算，没有赋值时的结果存储变量
eps	MATLAB计算浮点数的精确度
realmax	用户电脑能表示的最大正浮点数
realmin	用户电脑能表示的最小正浮点数
pi	圆周率π
i,j	虚数单位
inf	无穷大∞
NaN	不是一个数，例如0/0,inf/inf
flops	统计该工作空间中浮点数的计算次数
version	显示MATLAB版本

1.4 命令行的编辑和输入

启动MATLAB后，你就可以利用MATLAB工作。由于MATLAB采用的工作方式是一种交互式，随时输入命令，MATLAB便给出运算结果。

【例1.4-1】 计算$\frac{6\tan(0.13\pi)}{1+\sqrt{2}+\lg 5}$的值。

在MATLAB工作窗中键入：6 * tan(0.13 * pi)/(1+sqrt(2)+log10 (5))，然

后按 Enter 键，便可得出下面结果：

```
ans=
    0.83
```

若输入超过一行，可用续行符“...”。

例如计算1到10的算术和，可键入下面语句：

```
% 计算1到10的算术和
s=1+2+3+4+5+6+7...
  +8+9+10;
```

注意：MATLAB和C语言相似，对变量及命令有大小写形式之分。这里分号“;”表示不显示运算结果，而“%”是注释语句，计算机并不执行该语句，但通过help命令可把整个语句显示出来。

1.5 数据显示格式命令

在MATLAB中，计算出的数据可通过FORMAT命令控制数据显示格式。

【例1.5-1】 控制数据显示格式format。

```
a=4/3;
format long
a=
  1.33333333333333
```

其他格式的用法如表1-3。

表1-3 **数据显示格式命令format**

命令格式	显示情况	举例说明
format 或 format short	以5位浮点表示	1.3333
format long	以15位浮点表示	1.33333333333333
format short e	以5位浮点指数形式表示	1.3333e+000
format long e	以15位浮点指数形式表示	1.33333333333333e+000
format rat	以分数形式表示	4/3
format hex	以十六进制表示	3ff5555555555555
format bank	以(金融)元、角、分表示	1.33

如果采用菜单的话，则点击菜单File下的Prefrences选项，结果相同。

1.6 命令窗基本操作命令

在利用MATLAB工作之前，必须掌握MATLAB提供的基本键盘控制命令，他们可帮助你快速浏览MATLAB全部功能以及自由地在MATLAB内工作。

1.6.1 入门演示命令demo

MATLAB为用户提供了一套集成环境下的演示程序，图文并茂，可帮助用户尽快了解MATLAB的结构和功能。用户只需在命令窗上键入：

```
demo
```

回车后，用户在出现的对话框内通过选择不同的选项，便可看到MATLAB的全部内容。

1.6.2 内存变量管理

1. 储存WORKSPACE

MATLAB允许用户将内存中的变量，存储到一个文件里，自动形成扩展名为.mat文件，供以后使用。

例如，在命令窗上键入：

```
save data              %将内存全部变量存入data.mat中
save datax x           %将内存变量x存入datax.mat中
save datax x -ascii    %以ASCII格式，将内存变量x存入datax.mat中
save datay y -mat      %以二进制文件格式，将内存变量y存入datay.mat中
save datay x -append   %将内存变量x追加到datay.mat中
```

当然，也可以选择File菜单下的Save Workspace As…来进行储存。

2. 载入WORKSPACE

数据文件形成后，反过来还可调入内存。

```
load data          %将data.mat存储的内容、变量调入内存
load datay -mat    %以二进制文件格式，将datay.mat调入内存
```

也可以选择File菜单下的Load Workspace...把变量载入内存。

3. 查看WORKSPACE

当完成一个计算后，在内存中便会存在一些变量。我们通过命令whos或who可查看变量的情况。

```
whos
Name      Size       Bytes    Class
b         1×1            8    double array
```

```
y          1×1          8     double array
Grand total is 2 elements using 16 bytes
```

若选择File菜单下的Show Workspace，可查看内存变量。

4. 删除内存中变量

clear　　　　删除内存中全部变量。

若选择File菜单下的Show Workspace，然后选一变量，再选delete键，可删除该变量。

1.6.3 搜索路径

1. 改变路径命令cd

如果你已经在d:\stu下建立了一个文件夹zhang或目录，并且准备在里面工作，将以后形成的文件保存在此目录中，即可键入：

```
cd  d:\stu\zhang
```

2. 显示MATLAB整个系统设置的搜索路径path

如果你想设置搜索路径，可键入：

```
path(payh ,'d:\stu\zhang')
```

这样，MATLAB可自动搜索你所设置的路径d:\stu\zhang。

3. 路径浏览器pathtool

通过菜单File下的Set Path选项，弹出一个对话框——路径浏览器，填好你要设置的搜索路径d:\stu\zhang，再点击Add to Path按钮，同样可设置搜索路径。

1.6.4 在线帮助

如果你对某个函数或命令的定义及用法不太熟悉，就可用在线帮助命令help。

例如查一下sin函数的用法：

```
help sin
```

则MATLAB把注释语句显示如下：

```
SIN Sine.
    SIN(X) is the sine of the elements of X.
Overloaded methods
  help sym/sin.m
```

习 题 一

1. 用demo命令，熟悉MATLAB整体功能、用途和结构。
2. 在MATLAB命令窗口中，键入下列命令，进行观察，并考虑为什么？

```
a=1
b=2
c=a+b
whos
save aa c
dir
clear
whos
load aa
whos
```

3. 用在线帮助命令help，熟悉表1-1所有命令的功能及用途。
4. 用cd，path和pathtool三种命令，进入或设置你所在的工作环境。

第二章　数值计算

数值计算在现代科学研究中举足轻重，在MATLAB环境中进行数值计算就像在演算纸上算题一样非常方便，既解决了手工计算的痛苦，又避免了编程的烦恼。本章主要是高等数学、线性代数、数值分析中的常见问题：矩阵及矩阵的行列式值、迹、秩、条件数和范数；多项式运算；差分和导数；插值与拟合；特征值分解；矩阵的分解；非线性方程的解；数值积分；常微分方程的数值解。所以本章可作为工程计算的基础及数学类相关课程的实验手段。

2.1　向量和矩阵的构造

在MATLAB中，向量与矩阵的构造和编辑是极为方便的，不必像其他高级计算机语言那样要编写冗长的循环语句。

2.1.1　向量的生成

MATLAB中的"："很像其他高级计算机语言中的"TO"，有从初值到终值的含义。若遇到三目元素，则中间的是步长。

1. 由冒号运算符生成向量

【例2.1.1-1】 生成一个从0到π的行向量，步长为π/4=0.7854。

```
y=0:pi/4:pi
y=
  [0 0.7854 1.5708 2.3562 3.1416]
```

2. 由线性等分函数生成向量

可在首尾两端元素之间，等分建立新的向量。

(1) linspace(n1,n2)　包括n1,n2元素，生成100维向量；

(2) linspace(n1,n2,n) 包括n1,n2元素，生成n维向量。

【例2.1.1-2】 生成一个1～10增量为2的行向量。

```
x=linspace(1,10,2)
```

2.1.2　矩阵的生成

【例 2.1.2-1】　建立一个矩阵 $\begin{pmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{pmatrix}$。

```
x=[1 2 3; 4 5 6; 7 8 9]
x=
  1   2   3
  4   5   6
  7   8   9
```

这里每个元素必须用逗号","或空格" "隔开,每行元素则用分号";"隔开。

1. 由多个向量生成一个矩阵

首先要产生行向量,然后进行转置,再利用所得的列向量计算出它的函数向量,即可合成有两列的矩阵。

【例 2.1.2-2】　由 x,y 向量构造一个矩阵。

```
x=(0:0.2:1.0)';
y=exp(-x).*sin(x);
z=(1:size(x))'
[x y z]
ans =
         0         0    1.0000
    0.2000    0.1627    2.0000
    0.4000    0.2610    3.0000
    0.6000    0.3099    4.0000
    0.8000    0.3223    5.0000
    1.0000    0.3096    6.0000
```

2. 由下标编辑矩阵

向量矩阵元素的下标是非常重要的。我们可以对各元素很方便地进行编辑、替换、赋值等操作。

(1) 下标

【例 2.1.2-3】　用下标修改矩阵元素。

```
A=[1 2 3;
   4 5 6;
   7 8 9]
```

其中元素 A(3,3)=9,A=(1,3)=3,A(3,1)=7 等等。若用语句 A(3,3)=A

(1,3)+A(3,1),利用原矩阵的元素产生新元素(即为A(3,1)+A(1,3)=10)替代A矩阵中第三行第三列的元素A(3,3),则产生的新的A矩阵为：

```
A=[1 2 3;
   4 5 6;
   7 8 10]
```

(2) 使用“：”代替下标

【例2.1.2-4】 若有矩阵

```
A=[1 2 3;
   4 5 6;
   7 8 9]
```

结果会将A矩阵的2到3行、1和3列元素赋给B1矩阵：

```
   B1=A(2:3,[1 3])
B1 =
   4   6
   7   9
```

同理,将A矩阵的1和3行所有列元素赋给B2矩阵：

```
   B2=A([1 3],:)
B2 =
   1   2   3
   7   8   9
```

将2维全1矩阵赋给B3矩阵的3和1行、1和3列元素,其余元素为零：

```
   B3([3 1],[1 3])=ones(2)
B3 =
   1   0   1
   0   0   0
   1   0   1
  B4=A(:)'
```

将A矩阵所有行元素和列元素的转置赋给B4矩阵：

```
B4 =
   1   4   7   2   5   8   3   6   9
```

(3) 利用矩阵标识块进行赋值:X(m1:m2,n1:n2)=A,生成大矩阵

【例2.1.2-5】 *A* 阵同【例2.1.2-2】利用矩阵标识块生成 *X* 阵。

```
   X=zeros(4);    %定维
   X(2:4,1:3)=A
```

生成的($m2\times n2$)维X矩阵除赋值子矩阵和原已存在的元素外,其余元素都取零。

```
X =
    0    0    0    0
    1    2    3    0
    4    5    6    0
    7    8    9    0
```

3. 利用小矩阵组合大矩阵

【例 2.1.2 6】 由小矩阵组成大矩阵。

```
    a=[1 2;
       5 6 ];
    b=[3 4;
       7 8];
    c=[9 10;
       13 14];
    d=[11 12;
       15 16];
    s=[a b;
       c d]
s=
    1    2    3    4
    5    6    7    8
    9   10   11   12
   13   14   15   16
```

4. 细胞矩阵的生成

在细胞矩阵中，可将数据类型、维数皆不相同的元素，组成一个大矩阵。

【例 2.1.2-7】 利用【例 2.1.2-6】数据，生成细胞矩阵。

```
    S={a b; c d}         %花括号和向量中的方括号[ ]作用相似，表示细胞矩阵
      celldisp(S)        %显示细胞矩阵每个元素
      cellplot(S)        %用图形显示细胞矩阵
    %或用下面方法
    S=cell(2,2)          %cell 为细胞矩阵定维
    S(1,1)={a};S(1,2)={b};S(2,1)={c};S(2,2)={d};
    celldisp(S)
```

5. 利用“0-1”向量的下标

利用关系运算，建立“0-1”向量，然后对某一向量或矩阵进行过滤。MATLAB的这一独特方法，对后面构造和编辑矩阵来说是非常方便的。

【例 2.1.2-8】 利用“0-1”向量构造新向量。

```
    A=[1 2 3 4 5 6 7 8 9]
```

```
    L=A<=5          %小于等于5的元素为1,否则为0
    B=A(L)          %获得元素都不超过5的新向量
L =
    1   1   1   1   1   0   0   0   0
B =
    1   2   3   4   5
```

2.1.3 空 矩 阵

我们常常在命令窗上键入 x=[ ],然后回车,则会在内存产生一变量 x,它与 x=0结果不同,因为它的存储内容并不为零,仅仅为空;但它又和 clear x 或没输入不一样,若用 who x 检查,它又是存在的。

【例 2.1.3-1】 将 A 阵 $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \\ 16 & 17 & 18 & 19 & 20 \end{pmatrix}$ 的 2,5 列上的所有元素去掉,并分析空阵对向量或矩阵的过滤作用。

```
    A=[1 2 3 4 5;6 7 8 9 10 ;11 12 13 14 15;16 17 18 19 20]
    A(:,[2 5])=[]       %将2,5列上的所有元素去掉
A =
    1   3   4
    6   8   9
    11  13  14
    16  18  19
```

2.1.4 稀 疏 矩 阵

在工程计算中,许多矩阵都含有大量的零元素,例如有限元计算,称为稀疏矩阵。这些零元素往往占据大量的空间,影响矩阵生成和计算速度。为了提高速度,我们只存储非 0 元素,如采用等宽存储、一维存储等等。表 2-1 列出了 MATLAB 的有关稀疏矩阵函数,这些均放在\matlab\toolbox\sparfun 子目录下。

1. 稀疏矩阵的存储方式

在 MATLAB 中有两种存储矩阵的方式:一种是全元素(full)存储;另一种是稀疏(sparse)存储。实现两种存储方式如下:

sparse(A)　把矩阵存储方式从任何一种形式(含全元素形式)转变为稀疏形式。

full(A)　把矩阵存储方式从任何一种形式(含稀疏形式)转变为全元素形式。

表 2-1　　**MATLAB 的有关稀疏矩阵函数**

类　型	函数名	功　　能
稀疏矩阵的运算	issparse	检验稀疏矩阵,满足时为"1",否则为"0"
	nnz	求矩阵的非 0 元素总数
	nonzeros	求矩阵的非 0 元素数值
	nzmax	指定存放非 0 元素所需内存
	spalloc	为非 0 元素配置内存
	spfun	求各非 0 元素的函数值
	spones	用 1 置换非 0 元素
	condest	估计范-1 条件数
	normest	估计 2-范数
	sprank	结构秩
	gplot	依图论法则,画"(无向)图"
	spy	画稀疏结构图
基本稀疏矩阵生成	speye	生成单位稀疏阵
	sprand	生成均匀分布随机稀疏阵
	sprandn	生成正态分布随机稀疏阵
	spandsym	生成对称随机稀疏阵
转换函数	sparse	用向量生成稀疏阵
	spdiags	用带状向量生成稀疏阵
	find	查找非 0 元素的下标
	full	把稀疏阵转换成满矩阵
	spconvert	将外部数据格式转换成稀疏阵

2. 稀疏矩阵生成:sparse

SP=sparse(I, J, S, m, n, nzmax)

[I, J, S]这三个向量的长度相同,S 是按列排列的所有非零元素构成的主对角线向量;I,J 分别是非零元素的行下标和列下标向量;m,n 分别是生成稀疏矩阵

SP 的行、列维;nzmax 是用来为非零元素指定存储空间的正整数(它一定不小于非零元素总数)。

调用格式中的 6 个输入变量在一定条件下可以缺省。

如果 S = sparse(i,j,s,m,n),则将使 nzmax = length(s);

如果 S = sparse(i,j,s),则 m = max(i), n = max(j);

如果 S = sparse(m,n),则简化成 SPARSE([],[],[],m,n,0)。;

如果 S = sparse(A),则可变为[i,j,s] = find(A);S = sparse(i,j,s)。

【例 2.1.4-1】 将矩阵 $\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \\ 4 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ 变成稀疏矩阵。

```
A=[0 1 0; 0 0 2; 0 0 0; 4 3 0; 0 0 0];
SP=sparse(A)
SP =
   (4,1)     4
   (1,2)     1
   (4,2)     3
   (2,3)     2
```

【例 2.1.4-2】 构造一个带状矩阵 $\begin{bmatrix} -4 & 1 & & \\ 1 & -4 & 1 & \\ & 1 & -4 & 1 \\ & & 1 & -4 \end{bmatrix}$。

```
n=4;
SP=sparse(1:n,1:n,-4*ones(1,n),n,n,n); % 计算主对角线各元素
% 计算主对角线下面一根对角线各元素
SPL=sparse(2:n,1:n-1,ones(1,n-1),n,n,n-1);
SPU=SPL
SS=SP+SPL+SPU          % 计算三条对角线各元素
SF=full(SS)
```

3. 带状稀疏矩阵生成:spdiags

SP=spdiags(B,d,m,n)

m,n 分别指矩阵 SP 的行、列维。d 是长度为 p 的整数向量,它指定SM 的对角线位置;B 是全元素阵,用来生成 SP 的对角线位置上的元素。

【例2.1.4-3】 构造一个如同【例2.1.4-2】的带状矩阵$\begin{pmatrix} -4 & 1 & & \\ 1 & -4 & 1 & \\ & 1 & -4 & 1 \\ & & 1 & -4 \end{pmatrix}$。

```
n=4;
p=ones(n,1);
% [p,-4*p,p]三条对角线元素,[-1,0,1]三条对角线元素位置
SP2=spdiags([p,-4*p,p],[-1,0,1],n,n);
SF=full(SP2)
```

4. **外部数据转换为稀疏矩阵:spconvert**

SM=spconvert(T)

T来自外部数据文件(.mat文件或是.dat文件),数据文件.mat由load指令装载于MATLAB内存空间。

T数组的行维为非零元素数;对于实数列维为3,对于复数列维则为4。T数组的每一行指定一个稀疏矩阵元素,其前两列为位置下标向量。

【例2.1.4-4】 将矩阵$\begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \\ 4 & 4 & 4 \\ 5 & 5 & 5 \end{pmatrix}$变成稀疏矩阵。

```
T=[1 1 1; 2 2 2; 3 3 3; 4 4 4; 5 5 5]
SP=spconvert(T)
SP=
   (1,1)     1
   (2,2)     2
   (3,3)     3
   (4,4)     4
   (5,5)     5
```

2.1.5 多维矩阵

上面的矩阵都是二维矩阵,如果每个二维矩阵看成是一个矩阵页面,那么多个页面二维矩阵就是多维矩阵。例如,若想生成一个三维矩阵,就要生成多个二维矩阵$A(m,n,1),A(m,n,2),\cdots$

A(m,n,s), m,n可以是任意大于0的整数。

【例2.1.5-1】 三维矩阵的生成。

```
A(:,:,1) =[1 1 1 ;1 1 1 ;1 1 1]
A(:,:,2) =[2 2 2 ;2 2 2 ;2 2 2]
```

MATLAB 提供了多维矩阵的操作函数：

cat(dim,A1,A2,...)	矩阵组合；
ndims(A)	检查矩阵的维数；
ndgrid(x1,x2,x3)	生成多维 grid 向量；
repmat(A,[M N P])	生成同元素 A 多维矩阵。

【例 2.1.5-2】 三维矩阵的函数生成方法。

```
a=repmat(3,[2,3,2]);
b=cat(3,3*ones(2,3),3*ones(2,3)); %a 和 b 相同,生成方法不同
ndims(a)
a(:,:,1) =
    3    3    3
    3    3    3
a(:,:,2) =
    3    3    3
    3    3    3
ans =
    3
```

2.2 各种矩阵介绍

MATLAB 常见矩阵：单位阵、全 1 阵、全 0 阵、随机阵，它们均可产生($n\times n$)维方阵、($m\times n$)长方阵、与某阵 A 同维的矩阵，例如单位阵：eye(n)；eye(m,n)；eye(size(A))。

2.2.1 单位阵、全 1 阵和全 0 阵

函数	功能
eye	产生单位阵
ones	产生全 1 阵
zeros	产生全 0 阵

2.2.2 随 机 阵

1. 在[0,1]区间的均匀分布随机数

函数	功　　能
rand	产生[0,1]区间均匀分布随机阵
rand('seed',a)	令a为随机发生器的种子
rand('seed')	获取随机发生器的当前种子值

2. 服从 $N(0,1)$ 分布的正态随机数

函数	功　　能
randn	产生正态随机阵
randn('seed',a)	令a为随机发生器的种子
randn('seed')	获取随机发生器的当前种子值

【例2.2.2-1】 rand(n)与randn(n)的区别。

```
    rand(4)      %在[0,1]区间的均匀分布
    randn(4)     %服从正态分布
ans =
    0.9501    0.8913    0.8214    0.9218
    0.2311    0.7621    0.4447    0.7382
    0.6068    0.4565    0.6154    0.1763
    0.4860    0.0185    0.7919    0.4057
ans =
   -0.4326   -1.1465    0.3273   -0.5883
   -1.6656    1.1909    0.1746    2.1832
    0.1253    1.1892   -0.1867   -0.1364
    0.2877   -0.0376    0.7258    0.1139
```

【例2.2.2-2】 在一个4×4矩阵中，小于等于0.5的随机数有多少个？

```
    rand('seed',0)           %使随机发生器的种子为初始状态，具有可重复性
    x=rand(4)                %产生4×4矩阵
    N05=sum(sum(x<=0.5))     %先按列求出"0-1"向量，即<=0.5的所有元素的%
                             个数，然后按行求和
```

2.2.3 特 殊 阵

函 数	功 能	函 数	功 能
compan	伴随阵	diag	对角阵
hilb	Hilbert 阵	invhilb	hilbert 逆阵
hankel	Hankel 阵	kron	kronecker 张量阵
magic	魔方阵	pascal	pascal 阵
cross	矢量叉乘	dot	矢量点乘

1. 魔方阵

一个魔方阵的所有行、列、主次对角线的和是相同的。

【例 2.2.3-1】 求一个 3×3 的魔方阵。

```
    magic(3)
ans =
    8    1    6
    3    5    7
    4    9    2
```

2. Kronecker 张量阵

如果有 $A_{m\times n}$和 $B_{p\times q}$，则两矩阵的 Kronecker 积：

$$A_{m\times n}\quad B_{p\times q}=\begin{pmatrix} a_{11}\boldsymbol{B} & \cdots & a_{1n}\boldsymbol{B} \\ \vdots & & \vdots \\ a_{m1}\boldsymbol{B} & \cdots & a_{mn}\boldsymbol{B} \end{pmatrix}$$

【例 2.2.3-2】 求两矩阵的 Kronecker 积。

```
    A=[1 2 3; 4 5 6; 7 8 9]
    B=ones(2);
    c=kron(A,B)
    d=kron(B,A)
c =
    1    1    2    2    3    3
    1    1    2    2    3    3
    4    4    5    5    6    6
    4    4    5    5    6    6
    7    7    8    8    9    9
    7    7    8    8    9    9
```

```
d =
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
     1     2     3     1     2     3
     4     5     6     4     5     6
     7     8     9     7     8     9
```

3. 伴随阵

【例 2.2.3-3】 用三种方法生成多项式S^3+2S^2-5S+6的伴随阵。

```
CP=[1 2 -5 6]              % 多项式系数向量
A1=[-CP(2:4); eye(2) zeros(2,1)]
A2=-CP(2:4); A2(2:3,1:2)=eye(2)
A3=compan(CP)
CP =
     1     2    -5     6
A1 =
    -2     5    -6
     1     0     0
     0     1     0
A2 =
    -2     5    -6
     1     0     0
     0     1     0
A3 =
    -2     5    -6
     1     0     0
     0     1     0
```

4. Pascal 阵

【例 2.2.3-4】 对称、正定的Pascal阵。

```
pascal(3)
ans=
     1     1     1
     1     2     3
     1     3     6
```

5. 点乘和叉乘

【例 2.2.3-5】 矢量的点乘和叉乘。

```
a=[1 2 3];b=[3 2 1];
```

```
    dotb=dot(a,b)
    crossb= cross (a,b)
dotb=
      10
crossb =
         -4    8    -4
```

2.3 矩阵变换

2.3.1 矩阵的旋转和翻转

函数	功能
B=rot90(A)	将A阵逆时针方向旋转90度
B=rot90(A,k)	将A阵逆时针方向旋转(k×90)度,k为正负整数
B=fliplr(A)	将A阵左右翻转
B=flipud(A)	将A阵上下翻转

【例2.3.1-1】 矩阵的旋转和转置的区别。

```
    A=magic(3)    %魔方阵
    B90=rot90(A)
    BT=A'
    B180=rot90(A,2)
A =
    8    1    6
    3    5    7
    4    9    2
B90=
    6    7    2
    1    5    9
    8    3    4
BT =
    8    3    4
    1    5    9
    6    7    2
```

0959878

```
B180 =
     2     9     4
     7     5     3
     6     1     8
```

2.3.2 矩阵的变形

B＝reshape(A,m,n)可将任意行、列的A阵变成m×n的B阵，但元素的总数是不变的。

【例2.3.2-1】 用reshape和“:”的方法，将A向量变成$B_{3\times4}$阵和$B_{2\times6}$阵。

```
    A=1:6
    B=reshape(A,2,3)
    C=zeros(3,2);      %给C定维(3×2)
    C(:)=A(:)          %A以列的顺序赋给C变量
A =
     1     2     3     4     5     6
B =
     1     3     5
     2     4     6
C =
     1     4
     2     5
     3     6
```

2.3.3 矩阵对角线元素的提取

MATLAB可对某矩阵A在对角线方向提取元素形成新的向量或新的矩阵。

函　数	功　　能
diag(A,n)	提取A阵第n条对角线的元素。n为0,则为主对角线;n为正数,为A阵上数第n条对角线;n为负数,为A阵下数第n条对角线。
diag(A)	提取A阵主对角线的元素,与diag(A,0)相同。
tril(A,n)	提取A阵第n条对角线及以下的所有相应元素,其余为0。
tril(A)	提取A阵主对角线及以下的所有相应元素,其余为0,与tril(A,0)相同。
triu(A,n)	提取A阵第n条对角线及以上的所有相应元素,其余为0。
Triu(A)	提取A阵主对角线及以上的所有相应元素,其余为0,与triu(A,n)相同。

【例 2.3.3-1】 提取矩阵主对角线，及下一对角线以下、上一对角线以上所有相应元素。

```
A=[1 2 3 4 5;6 7 8 9 10; 11 12 13 14 15]
z=diag(A)'           %提取 A 阵主对角线
AL=tril(A,-1)        %提取 A 阵主下一对角线以下所有相应元素
AU=triu(A,1)         %提取 A 阵主上一对角线以上所有相应元素
Z=diag(z)
A =
    1    2    3    4    5
    6    7    8    9   10
   11   12   13   14   15
z =
    1 7    13
AL =
    0    0    0    0    0
    6    0    0    0    0
   11   12    0    0    0
AU =
    0    2    3    4    5
    0    0    8    9   10
    0    0    0   14   15
Z =
    1    0    0
    0    7    0
    0    0   13
```

2.4 运算符及运算规则

在算术、关系、逻辑运算中，运算优先权按顺序排列：算术、关系、逻辑运算。

2.4.1 算术运算

在 MATLAB 里进行计算非常方便，只要你在命令窗上输入一个算式，就会得出答案。

```
1+2+3
ans=
    6
```

算术运算可参考表 2-2。

表 2-2 **算术运算符及算例**

算术运算	算术运算符	示　　例
加法，$a+b$	+	2+3
减法，$a-b$	-	54-17
乘法，$a\times b$	*	3.14*0.36
除法，$a\div b$	/或者\	63/9=9\63
乘方，a^b	^	8^2

2.4.2 关系运算符及关系表达式

关系运算符：

运算符	符号说明
<	小于
<=	小于等于
>	大于
>=	大于等于
==	等于
~=	不等于

关系表达式是由关系运算符将两个常量或变量连接的表达式。例如：$a>3$，$2>3$，$a<b$ 等等。

(1) 若关系运算符两端的比较量是标量：当关系成立时，则返回 1；当关系不成立时，则返回 0。例如：

```
    2>3
ans=
    0
```

(2) 当参与比较的是维数相同的数组或矩阵 A 和 B 时，则每个元素逐个比较，结果为维数相同的由"0"和"1"组成的数组。

【例 2.4.2-1】 一个随机阵与数 0.5、全 0.5 阵的比较。

```
A=rand(4);
x=0.5;
Xx=0.5*ones(4)
Ax=A<=x          %矩阵与一个数的比较
Axx=A<=Xx        %矩阵 A 与全 0.5 阵的比较后生成 0-1 阵，可以提取需要的数据
```

和信息

```
Xx =
      0.5000    0.5000    0.5000    0.5000
      0.5000    0.5000    0.5000    0.5000
      0.5000    0.5000    0.5000    0.5000
      0.5000    0.5000    0.5000    0.5000
Ax =
      0    0    0    0
      1    0    1    0
      0    1    0    1
      1    1    0    1
Axx =
      0    0    0    0
      1    0    1    0
      0    1    0    1
      1    1    0    1
```

2.4.3　逻辑运算符及运算规则

运算符：

运算符	说明
&	与
|	或
~	非

运算规则：

(1) 在逻辑运算中，非零元素的逻辑量为“真”，用“1”表示；零元素的逻辑量为“假”，用“0”表示。

(2) 当比较量是标量，a，b 间关系成立，则

a	b	$A\&b$	$a\|b$	$\sim a$
1	1	1	1	0
0	1	0	1	1
1	0	0	1	0
0	0	0	0	1

（3）当参与逻辑运算的是维数相同的数组或矩阵 **A** 和 **B** 时，则 **A** 将对 **B** 位置相同的诸元素按规则(2)逐个进行运算。结果是生成与 **A**(或 **B**)同维的数组，其元素均由"1"或"0"组成。

【例 2.4.3-1】 矩阵的逻辑运算。

```
    A=fix(10*(rand(3)))
    B=[-1 0 0;0 0.5 0; 0 0 1];
    AandB=A&B
    AorB=A|B
    NotB=~B
A =
    9    8    8
    9    0    0
    4    3    1
AandB =
    1    0    0
    0    0    0
    0    0    1
AorB =
    1    1    1
    1    1    0
    1    1    1
NotB =
    0    1    1
    1    0    1
    1    1    0
```

2.5 矩 阵 运 算

2.5.1 矩 阵 转 置

【例 2.5.1-1】 将行向量变成列向量。

```
    [-2 1 5]'
ans=
    -2
     1
     5
```

【例 2.5.1-2】 求$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$的转置。

```
    A=[1 2 3;4 5 6;7 8 9]'
A =
    1    4    7
    2    5    8
    3    6    9
```

2.5.2 矩阵乘法

在 MATLAB 中,矩阵的乘、除、乘方及各种函数运算等均有两种方式:

1. 矩阵运算;
2. 元素运算(. 运算)。

故在乘法运算中,有下面相应的两种运算:

1. 矩阵乘 (*):两个内维相同的矩阵相乘。
2. 元素乘(. *):两个同维的矩阵相应元素相乘。

【例 2.5.2-1】 比较矩阵的两种乘法。

```
    A=[1 2 3;
       4 5 6;
       7 8 9]
    B=eye(3)
    C=A * B            %矩阵乘
    D=A. * B           %元素乘(·乘)
A =
    1    2    3
    4    5    6
    7    8    9
B =
    1    0    0
    0    1    0
    0    0    1
C =
    1    2    3
    4    5    6
    7    8    9
D =
```

```
1   0   0
0   5   0
0   0   9
```

2.5.3 矩阵除法、逆与线性方程的解

矩阵的除运算和乘一个矩阵的逆在线性代数的解线性方程运算中是相同的，而且前者在数学界常常被认为概念是错误的。但是，在MATLAB系统中，运算是不同的，矩阵除运算速度要快些。

1. 矩阵除

在MATLAB中，矩阵除有两种方式：

(1) 矩阵除(左除"\"或右除"/")：

MATLAB左除"\"，例如：$\boldsymbol{A}\backslash\boldsymbol{B}$，那么方程 $\boldsymbol{X}=\boldsymbol{A}\backslash\boldsymbol{B}$ 则是 $\boldsymbol{A}*\boldsymbol{X}=\boldsymbol{B}$ 的解。

MATLAB右除"/"，例如 $\boldsymbol{B}/\boldsymbol{A}$，那么方程 $\boldsymbol{X}=\boldsymbol{B}/\boldsymbol{A}$ 则是 $\boldsymbol{X}*\boldsymbol{A}=\boldsymbol{B}$ 的解。

(2) 元素除(元素左除".\"或元素右除"./")：两个同维的矩阵相应元素相除。

$$b.\backslash\boldsymbol{A}=\begin{bmatrix} b\backslash a_{11} & \cdots & b\backslash a_{1n} \\ \vdots & & \vdots \\ b\backslash a_{n1} & \cdots & b\backslash a_{nn} \end{bmatrix}$$

$$\boldsymbol{B}.\backslash\boldsymbol{A}=\begin{bmatrix} b_{11}\backslash a_{11} & \cdots & b_{1n}\backslash a_{1n} \\ \vdots & & \vdots \\ b_{n1}\backslash a_{n1} & \cdots & b_{nn}\backslash a_{nn} \end{bmatrix}$$

2. 矩阵的逆

如果 $\boldsymbol{B}$ 阵为单位阵，方程 $\boldsymbol{A}*\boldsymbol{X}=\boldsymbol{I}$ 的解就成为了矩阵 $\boldsymbol{A}$ 的逆，即：

inv($\boldsymbol{A}$)

3. 解线性方程

对于方程 $\boldsymbol{AX}=\boldsymbol{B}$，其中 $\boldsymbol{A}$ 是 $(m\times n)$ 的矩阵。

(1) 当 $n=m$ 且非奇异时，此方程为"恰定"方程组。

(2) 当 $n>m$ 时，此方程为"超定"方程组。

(3) 当 $n<m$ 时，此方程为"欠定"方程组。

MATLAB解恰定方程 $\boldsymbol{A}*\boldsymbol{X}=\boldsymbol{B}$ 的方法：

(1) 采用求逆运算解方程

```
x=inv(A)*B
```

(2) 采用左除运算解方程

```
x=A\B
```

【例 2.5.3-1】 "求逆"法和"左除"法解恰定方程$\begin{cases}10x_1+x_2+x_3=12\\2x_1+10x_2+x_3=13\\2x_1+2x_2+10x_3=14\end{cases}$

```
A=[10 1 1; 2 10 1; 2 2 10];
B=[12 13 14]';
format long
XI=A\B              %XI是用"左除"法解恰定方程所得的解
XD=inv(A)*B         %XD是用"求逆"法解恰定方程所得的解
con=cond(A)
```

这里如果 A 的条件数 cond(A)不大的话，则原始资料对解的影响也不大，呈良性，否则方程解为病态。参见 2.11.3 节。

```
XI =
    1.00000000000000
    1.00000000000000
    1.00000000000000
XD =
    1.00000000000000
    1.00000000000000
    1.00000000000000
con =
    1.53218187810987            %呈良性
```

MATLAB 解超定方程 $A*X=B$ 的方法：

【例 2.5.3-2】 用左除法计算超定方程。

```
a=[1 2 3 ;4 5 6; 7 8 9;10 11 12];
b=[1 2 3 4];
x=a\b
x=
    0.0000
         0
    0.3333
```

2.5.4 矩阵的伪逆

当矩阵 A 为长方阵时，方程 $AX=I$ 和 $XA=I$ 至少有一个无解。我们想找一个矩阵，在某种意义上代替矩阵的逆，就是伪逆。

命令格式：

X=pinv(A)

【例 2.5.4-1】 求一个矩阵 A 的伪逆。

```
    A=[ 1 2;3 4;5 6];
    X=pinv(A)
X =
    -1.3333   -0.3333    0.6667
      1.0833    0.3333   -0.4167
      I=X * A              %检验是否为单位阵
I=
      1.0000    0.0000
      0.0000    1.0000
      P=A * X              %但 P=A * X 不是单位阵
P =
      0.8333    0.3333   -0.1667
      0.3333    0.3333    0.3333
    -0.1667    0.3333    0.8333
```

注意：

```
    P * A              %P * A =A
ans =
    1.0000    2.0000
    3.0000    4.0000
    5.0000    6.0000
      X * P              %X * P=X
ans =
    -1.3333   -0.3333    0.6667
      1.0833    0.3333   -0.4167
```

这里 $P*A=A$ ，$X*P=X$，P 很类似单位阵，并且验证可得：$A*X*A=A$ ，$X*A*X=X$ 这就是伪逆 X 的定义。

2.5.5 矩阵乘方 A^b

1. 矩阵的乘方

一个矩阵 A 的乘方 A^b 可分为不同的情况。

当 b 是整数时：

(1) 如果 $b>0$，A ^ b 可看作方阵 A 自乘 b 次的结果。

(2) 如果 $b<0$，A ^ b 可看作方阵 A 的逆自乘 b 次的结果。

(3) 如果 $b=0$，A ^ 0 可看作与 A 阵同维的单位阵。

当 b 取非整数时：

若 A 阵有特性值 λ，可分解为 $AP=P\lambda$，其中 λ 为对角阵即特征值(特征根)矩阵，P 为特征向量矩阵。即满足：

$$A=P\lambda P^{-1} \text{或} A=P\begin{pmatrix}\lambda_{11} & & \\ & \ddots & \\ & & \lambda_{nn}\end{pmatrix}P^{-1}$$

则有：

$$A^b=P\begin{pmatrix}\lambda_{11}^b & & \\ & \ddots & \\ & & \lambda_{nn}^b\end{pmatrix}P^{-1}$$

【例 2.5.5-1】 求矩阵的整数乘方，并验证。

```
    A=fix(10*rand(3))
    AB=A^2
    AA=A*A  %验证上面AB结果
A =
     9     4     4
     2     8     0
     6     7     8
AB =
   113    96    68
    34    72     8
   116   136    88
AA =                    % AA =AB
   113    96    68
    34    72     8
   116   136    88
```

2. 矩阵 $A_{m\times n}$ 的 b 元素乘方(.^)

矩阵 $A_{m\times n}$ 的 b 元素乘方(.^)相当于矩阵 A 中每个元素的 b 乘方。即：

$$A.\hat{}\ b=\begin{pmatrix}a_{11}^b & \cdots & a_{1n}^b \\ \vdots & & \vdots \\ a_{m1}^b & \cdots & a_{mn}^b\end{pmatrix}$$

【例 2.5.5-2】 求【例 2.5.5-1】矩阵 A 的元素运算乘方。

```
    APB=A.^2
APB =                   %注意结果与例2.5.5-1的AB不同
    81    16    16
     4    64     0
    36    49    64
```

3. 一个数的矩阵乘方 b

一个数的矩阵乘方 b，相当于对式 $A=PLP^{-1}$ 中的特征值矩阵 L 的每个元素的矩阵乘方。即：

$$b^{A}=P\begin{bmatrix} b^{\lambda_{11}} & & \\ & \ddots & \\ & & b^{\lambda_{mn}} \end{bmatrix}P^{-1}$$

【例 2.5.5-3】 求一个数的矩阵乘方 2^{A}，矩阵 A 与【例 2.5.5-1】相同。

```
    y=2^A
y =
    1.0e+004 *
    1.4635    1.7193    0.8557
    0.4278    0.5094    0.2467
    1.7153    2.0217    1.0029
```

4. 一个数的元素乘方

一个数的元素乘方，相当于对该数进行矩阵每个元素的乘方。即：

$$b.\hat{}\ A=\begin{bmatrix} b^{a11} & \cdots & b^{a1n} \\ \vdots & & \vdots \\ b^{am1} & \cdots & b^{amn} \end{bmatrix}$$

【例 2.5.5-4】 求一个数的元素乘方 $2.^{A}$，矩阵 A 与【例 2.5.5-1】相同。

```
    bA=2.^A
bA =
    512    16    16
      4   256     1
     64   128   256
```

2.5.6　元素函数和矩阵函数

1. 元素函数

按元素运算法则计算，即 $f(A)=\begin{bmatrix} f(a_{11}) & \cdots & f(a_{1n}) \\ \vdots & & \vdots \\ f(a_{m1}) & \cdots & f(a_{mn}) \end{bmatrix}$，$f$ 是对矩阵进行元素函数计算。

2. 矩阵函数

若 A 阵有特性值分解 $AP=P\lambda$，其中 λ 为对角阵即特征值（特征根）矩阵，P 为特征向量矩阵，即满足：

$$A=P\lambda P^{-1}\text{或}A=P\begin{pmatrix}\lambda_{11} & & \\ & \ddots & \\ & & \lambda_{nn}\end{pmatrix}P^{-1}$$

MATLAB 命令格式：

[P,L]—cig(A)

注意：λ 特征根矩阵 MATLAB 用 L 表示，那么矩阵函数运算可表示为：

fm(A)=P * diag(fm(diag(L))) * P^{-1}

矩阵的超越函数要求运算矩阵必须为方阵。这里 fm 是对矩阵进行某种矩阵函数计算(如下表第二列)。

矩阵的两种超越函数运算对照：

元素函数指令	矩阵函数指令
exp(A)	expm(A)
log(A)	logm(A)
sqrt(A)	sqrtm(A)
	funm(A,'FN')

【例 2.5.6-1】 对矩阵 *A* 分别进行元素函数 sqrt 与矩阵函数 sqrtm 运算。

```
A=[9 4;16 25];
Ab=sqrt(A)
Am=sqrtm(A)
AbAb=Ab*Ab
AbAb1=Ab.*Ab
AmAm=Am*Am
Ab =
     3     2
     4     5
Am =
     2.8146    0.5191
     2.0764    4.8910
AbAb =
     17    16
     32    33
AbAb1 =
      9     4
```

```
    16     25
AmAm =
     9.0000     4.0000
    16.0000    25.0000
```

【例 2.5.6-2】 求 log(A)、logm(A)和 funm(A,'log')矩阵，A 与【例 2.5.6-1】相同。

```
logA=log(A)
logmA=logm(A)
funmA=funm(A,'log')
logA =
    2.1972    1.3863
    2.7726    3.2189
logmA =
    1.9731    0.2838
    1.1351    3.1083
funmA =
    1.9731    0.2838
    1.1351    3.1083
```

2.6 多 项 式

2.6.1 多项式的表达和建立

若有多项式 $P(x)=a_0+a_1x+\cdots+a_{n-1}x^{n-1}+a_nx^n$，则可用下面系数行向量表示：

$$\boldsymbol{P}=[a_0\ a_1\cdots\ a_{n-1}\ a_n]$$

多项式向量的创建函数：

P=poly(A)

可产生多项式系数向量 $\boldsymbol{P}$。

若 $\boldsymbol{A}$ 是向量，即 $\boldsymbol{A}=[b_1\ b_2\cdots\ b_n]$，则这个向量为多项式的根，即满足关系式：

$$(x-b_1)(x-b_2)\cdots(x-b_n)=a_0+a_1x+\cdots+a_{n-1}x^{n-1}+a_nx^n$$

若 $\boldsymbol{A}$ 是 n 阶方阵，则多项式为特征多项式，矩阵则为特征多项式的根。

$$P(\boldsymbol{A})=a_0\boldsymbol{I}+a_1\boldsymbol{A}+\cdots+a_{n-1}\boldsymbol{A}^{n-1}+a_n\boldsymbol{A}^n$$

【例 2.6.1-1】 求 3 阶方阵 $\boldsymbol{A}$ 的特征多项式。

```
A=[11 12 13;14 15 16;17 18 19]
```

```
PA=poly(A)              %产生多项式系数向量
PPA=poly2str(PA,'x')    %产生多项式关系式
```

2.6.2 多项式乘除运算

多项式乘除运算是控制系统、信号处理的重要数学工具。因为卷积、解卷和多项式的乘除运算机理相同。我们这里就使用卷积conv(a,b)和解卷deconv(a,b)进行多项式乘除运算。

1. 多项式乘运算

命令格式:c=conv(a,b)

【例2.6.2-1】 展开$(s^2+5s+6)(s+3)(s+2)$。

```
    c=conv([1,5,6],conv([1,3],[1,2]))
    C=poly2str(c,'s')
c =
    1    10    37    60    36
C =
    s^4 + 10 s^3 + 37 s^2 + 60 s + 36
```

2. 多项式除运算

解卷是卷积的逆运算。用向量 $\boldsymbol{a}$ 对向量 $\boldsymbol{c}$ 进行解卷将得到商向量 $\boldsymbol{q}$ 和余向量 $\boldsymbol{r}$。

命令格式: [q,r]=deconv(c,a)

【例2.6.2-2】 求【例2.6.2-1】所得向量 $\boldsymbol{c}$ 分别被$(s+2)$,$(s+5)$除后的结果。

```
    [q1,r1]=deconv(c,[1,2])
    [q2,r2]=deconv(c,[1,5])
q1 =
    1    8    21    18
r1 =
    0    0    0    0    0
q2 =
    1    5    12    0
r2 =
    0    0    0    0    36
```

说明【例2.6.2-1】中的$C=s^4+10s^3+37s^2+60s+36$不能被$(s+5)$整除,余项是常数36。

2.6.3 常见多项式运算

1. 求多项式向量的根

函数格式：r＝roots(p)，求多项式向量 p 的根。

【例 2.6.3-1】 求(x^3-3x^2-7x-2)的根。

```
    r=roots([1 -3 -7 -2])
r =
     4.6119
    -1.2705
    -0.3413
```

2. 求矩阵的特征多项式

函数格式：poly(A)，求矩阵 A 的特征多项式。

【例 2.6.3-2】 $A=\begin{pmatrix}1 & 2\\3 & 4\end{pmatrix}$，求矩阵的特征多项式。

```
    A=[1 2; 3 4];
    f=poly(A)
    FX=poly2str(f,'x')
f =
    1.0000   -5.0000   -2.0000
FX =
    x^2 - 5 x - 2
```

3. 求矩阵的特征多项式及其值

函数格式：Pq＝polyval(fx,a)，按元素运算规则计算特征多项式值。fx 为多项式，a 为一个数或向量。

函数格式：PQ＝polyvalm(FX,A)，按矩阵运算规则计算特征多项式值。FX 为多项式，A 为矩阵。

【例 2.6.3-3】 求【例 2.6.3-2】的特征多项式中 FX(10)和 FX(A)的值。

```
    Pq=polyval(f,10)
    Pq1=polyval(f,[1 2 3 4])
    PQ=polyvalm(f,A)
Pq =
    48
Pq1 =
    -6   -8   -8   -6
PQ =
    1.0e-015 *
     0.2220        0
     0.0000   0.2220
```

4. 多项式部分分式展开

[r,p,k]=residue(b,a)　b,a 分别是分子分母多项式系数向量,r,p,k 分别是留数、极数、直项。

对于多项式 $b(s)$ 与不含重根的 n 阶多项式 $a(s)$ 之比,可展开如下:

$$\frac{b(s)}{a(s)}=\frac{r_1}{s-p_1}+\frac{r_2}{s-p_2}+\cdots+\frac{b(s)}{a(s)}+k(s)$$

假如含有重根,那么相应部分则写为:

$$\frac{r_j}{s-p_j}+\frac{r_2}{s-p_2}+\cdots+\frac{r_{j+m-1}}{(s-p_j)^m}$$

【例 2.6.3-4】 $\frac{3x+7}{x^2+5x+6}$部分分式展开。

```
    a=[1,5,6];
    b=[3,7];
    [r,p,k]=residue(b,a)
r =
    2
    1
p =
    -3
    -2
k =
    [ ]
    [r2,p2,k2]=residue(r,p,k)      %还原其分式
r2 =
    3    7
p2 =
    1    5    6
k2 =
    [ ]
```

2.7 差分和导数

差分和导数在数值计算中非常重要。MATLAB 提供许多关于差分、导数及偏导的函数:

DX=diff(X,k)	按(m×n)维X阵的列求((m−k)×n)维k阶差分矩阵
dyx- diff(Y). /diff(X)	求差商,Y,X是同维矩阵
dyx=gradient(Y,dx)	求偏导,Y是向量,dx是X的导数
[GX,GY]=gridient(Z,dx,dy)	GX,GY分别是二元函数z关于x,y的偏导

【例2.7-1】 按列求差分。

```
    X=[1,2,3;4,5,6;7,8,9;10,11,12]
    D=diff(X)              %((4-1)×3)维1阶差分矩阵
    dyx=gradient(Y,0.8)
D =
    3    3    3
    3    3    3
    3    3    3
```

【例2.7-2】 对 x,y 求差商。

```
    x=[0 2 3 5 6]
    y :[0 8 27 125 216]
    dyx-diff(y). /diff(x)
dyx =
    4    19    49    91
```

【例2.7-3】 求二元函数偏导数(二次曲面的等高线和梯度)。

```
    x=-2:0.2:2;y=x;
    [X,Y]=meshgrid(x,y);        %用于构造曲面的栅格坐标
    R=sqrt(X.^2+Y.^2)+eps;
    Z=sin(R)./R;
    mesh(X,Y,Z);                %如图2.7-1所示
    pause
    clf;
    [GX,GY]=gradient(Z,0.3,0.3); % GX,GY分别是X和Y的偏导
    contour(X,Y,Z,'b'),    %等值线,如图2.7-2所示
    hold on,               %保持屏幕
    quiver(X,Y,GX,GY,'r'),   %矢量场图,小箭头表示梯度
    hold off
```

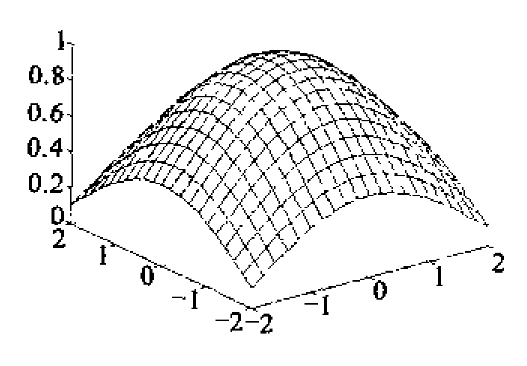


图 2.7-1 二次曲面

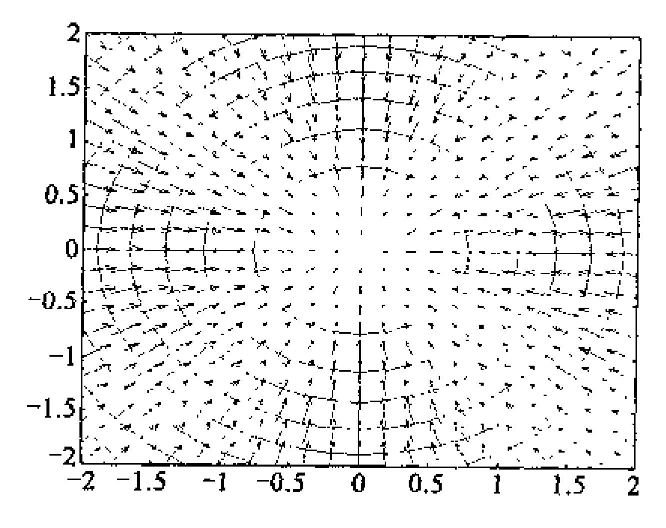


图 2.7-2 二次曲面的等高线和梯度

2.8 插值与拟合

2.8.1 插 值

插值是数值计算和工程计算的一个重要问题，可在已知的一些少数离散数据中，如 $a \leqslant x1 < x2 < \cdots < xn \leqslant b$，由不同的算法求出插值函数、两节点间的近似值及其向量。MATLAB 已提供了许多的插值函数如 INTERP1，INTERP1Q，INTERPFT，SPLINE，INTERP2，INTERP3，INTERPN 均放在 matlab\toolbox\matlab\polyfun 子目录下。

1. 一维插值

一维插值主要有三种方法：

函　数	说　明	用　法
INTERP1Q	快速插值法	F=INTERP1Q(X,Y,XI)
INTERP1	多项式插值法	YI = INTERP1(X,Y,XI,'method')
INTERPFT	FFT 的插值法	YI = INTERPFT(X,N)

注意:X,Y 为已知数据,XI 为需要插值的位置向量。

'method'共有四种方法选择：

'nearest'	最近点插值法	取较近点的值
'linear'	线性插值法	用直线连接数据点
'spline'	样条插值法	用三次样条曲线通过数据点
'cubic'	立方插值法	用三次曲线通过数据点

2. 多维插值

多维插值也有三种方法：

函　数	说　明	用　法
INTERP2	二维插值法	ZI = INTERP2(X,Y,Z,XI,YI,'method')
INTERP3	三维插值法	VI = INTERP3(X,Y,Z,XI,YI,ZI,'method')
INTERPN	多维插值法	VI = INTERPN(X,Y,Z,XI,YI,ZI,'method')

注意:同理,这里 X,Y,Z 为已知数据,XI,YI,ZI 为需要插值的位置向量。

'method'共有三种方法选择：

'nearest'	最近点插值法	取较近点的值
'bilinear'	双线性插值法	用双线性曲面连接数据点
'bicubic'	双立方插值法	用双三次曲面通过数据点

【例 2.8.1-1】 已知数据的列表如下：

x	−2	0	4	5
y	5	1	−3	1

(1) 用四种方法算出在 0.3 处的近似值。

```
    x=[-2 0 4 5];
    y=[5 1 -3 1];
    format rat
    yn=interp1(x,y,-1,'nearest')
    yl=interp1(x,y,-1,'linear')
    ys=interp1(x,y,-1,'spline')
    yc=interp1(x,y,-1,'cubic')
yn =
    1
yl =
    3
ys =
    24/7
yc =
    3
```

(2) 算出与 *xi* 向量相应的近似值 *yi* 向量，并画出曲线（如图 2.8.1-1）。

```
    xi=[-2:5]
    yl=interp1(x,y,xi,'linear')   %红色
    ys=interp1(x,y,xi,'spline')   %蓝色
    yc=interp1(x,y,xi,'cubic')    %黄色
    yf=interpft(x,max(size(xi)))
    plot(x,y,'o',xi,yl,'-r',xi,ys,'-b',xi,yc,'-y')
yl =
    0.2500    0.3750    0.5000    0.7500    1.0000    1.5000    2.0000    3.0000
    4.0000
```

【例 2.8.1-2】 对于已知数据的函数曲面用三种方法进行插值，并用图形显示结果。

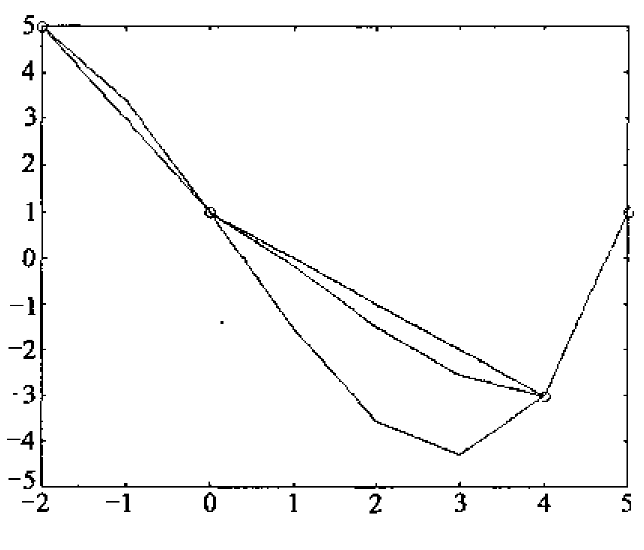


图 2.8.1-1 四种插值曲线

```
clf;
[x,y]=meshgrid(-4:4);
%z=peaks(x,y);
z=-4*x.^2-y.^2;
mesh (x,y,z);  %surf
title('绘制原数据曲面')  %图 2.8.1-2 所示
[xi,yi]=meshgrid(-4:0.25:4);
zin=interp2(x,y,z,xi,yi,'nearest') ;
zil=interp2(x,y,z,xi,yi,'bilinear');
zic=interp2(x,y,z,xi,yi,'bicubic');
pause;disp('请回车')
mesh (xi,yi,zin)  % surf
title('绘制最近点插值曲面')  %如图 2.8.1-3 所示
pause ;disp('请回车')
mesh(xi,yi,zil)  % surf
title('绘制双线性插值曲面')
pause;disp('请回车')
mesh(xi,yi,zic)
title('绘制双立方插值曲面')
```

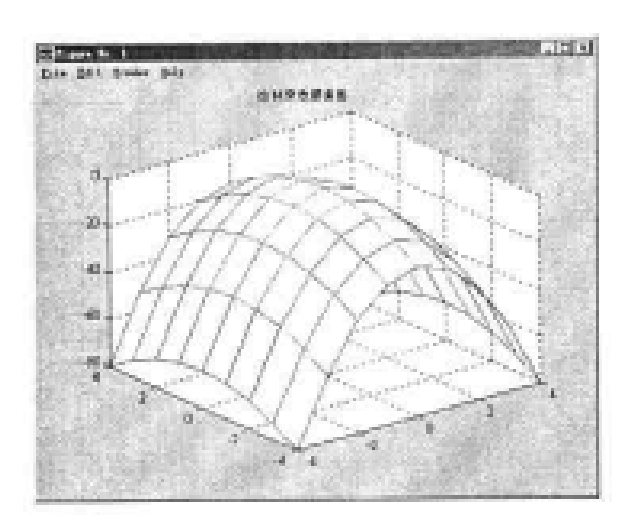


图 2.8.1-2　原数据曲面

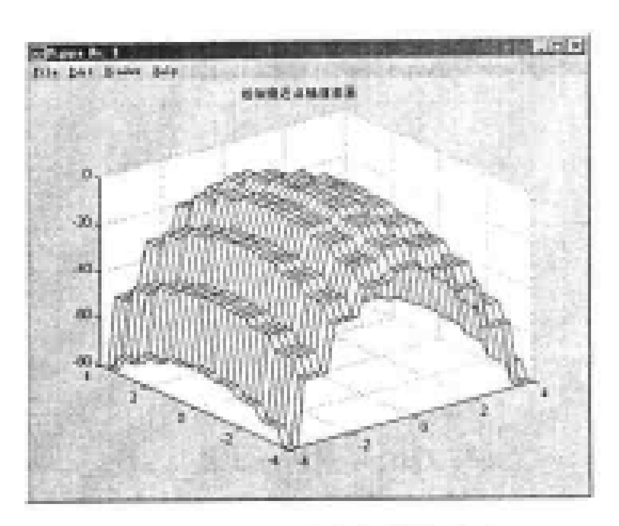


图 2.8.1-3　最近点插值曲面

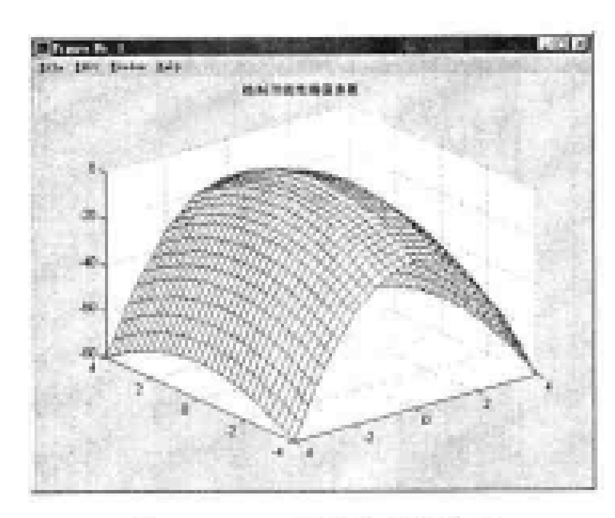


图 2.8.1-4　双线性插值曲面

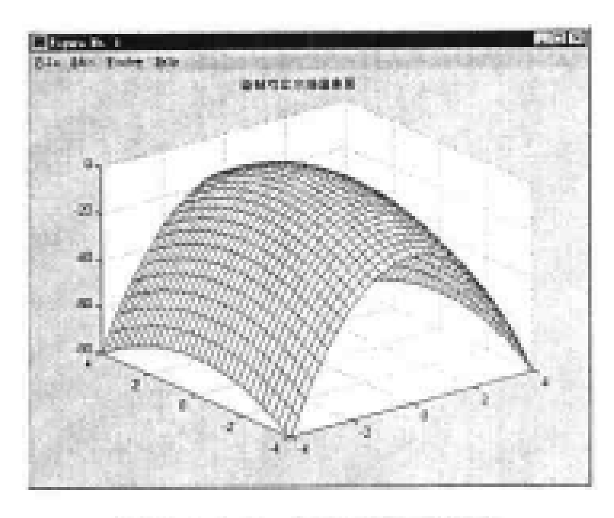


图 2.8.1-5　双立方插值曲面

2.8.2 拟　　合

在实验、测量或统计数据时，常常用一个拟合曲线或方程来逼近这些数据(x_i, y_i)，而要求与数据的误差平方和为最小。设拟合曲线为 $S(x)$，则误差：

$$R=\sum_{i=0}^{m}[S(x_i)-y_i]^2=\min$$

这就是曲线拟合的最小二乘法。

1. 用教科书常规方法拟合数据

【例 2.8.2-1】 求一拟合曲线，逼近下列数据：

i	0	1	2	3	4	5	6	7	8
X_i	1	3	4	5	6	7	8	9	10
Y_i	10	5	4	2	1	1	2	3	4

(1) 观察数据分布趋势(图 2.8.2-1)

```
xi=[1 3 4 5 6 7 8 9 10]
yi=[10 5 4 2 1 1 2 3 4]
plot(xi,yi,'o'),grid o'
```

(2) 根据画出的图选定方程 $y=a_0+a_1x+a_2x^2$

对于方程有 a_0,a_1,a_2 三个未知数,9 个线性方程,则

$$\boldsymbol{Y}=\begin{bmatrix}10\\5\\4\\2\\1\\1\\2\\3\\4\end{bmatrix}=a_0\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix}+a_1\begin{bmatrix}1\\3\\4\\5\\6\\7\\8\\9\\10\end{bmatrix}+a_2\begin{bmatrix}1^2\\3^2\\4^2\\5^2\\6^2\\7^2\\8^2\\9^2\\10^2\end{bmatrix}=\begin{bmatrix}1&1&1^2\\1&3&3^2\\1&4&4^2\\1&5&5^2\\1&6&6^2\\1&7&7^2\\1&8&8^2\\1&9&9^2\\1&10&10^2\end{bmatrix}\begin{bmatrix}a_0\\a_1\\a_2\end{bmatrix}=\boldsymbol{X}*a$$

(3) 建立 $\boldsymbol{X}$ 阵

```
X=[ones(size(xi)) ; xi ; xi.^2]
```

(4) 求方程解

```
a=X'\yi'
a =
    13.4597
   -3.6053
    0.2676
```

拟合曲线方程:$y=13.46-3.61x+0.27x^2$

```
f2=polyval(flipud(a),xi);   %按数组运算规则计算多项式值
plot(xi,yi,'bo',xi,f2,'r-')   %绘出已知数据和拟合曲线图
legend('原数据线','阶拟合曲线')   %图形注释,如图 2.8.2-2 所示
```

(5) 如果步骤(2)选定方程 ax^b,则

```
X=[ones(size(xi)) log(xi)];
```

```
    a=X\log(yi)
a =
      2.1257
     -0.6913
%代入ax^b得：y=exp(2.1257)*x^-0.6913
    yy=exp(2.1257)*xi.^(-0.6913);
    plot(xi,yi,'bo',xi,yy,'r-',xi,f2,'b-')    %绘图比较已知数据和拟合曲线图
    legend('原数据线','ax^b拟合曲线','2阶拟合曲线')
```

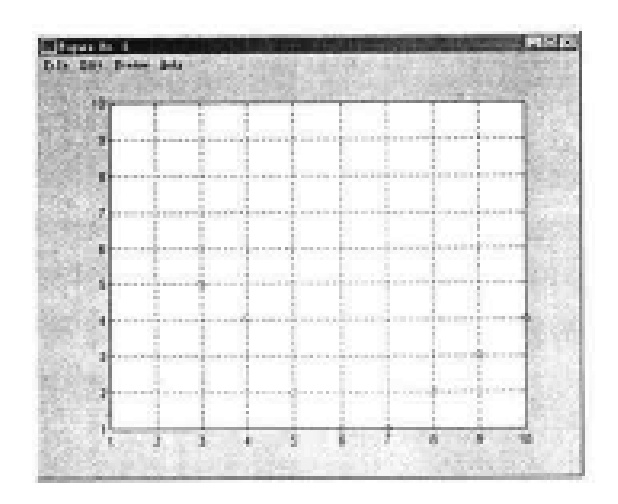


图2.8.2-1　数据分布趋势

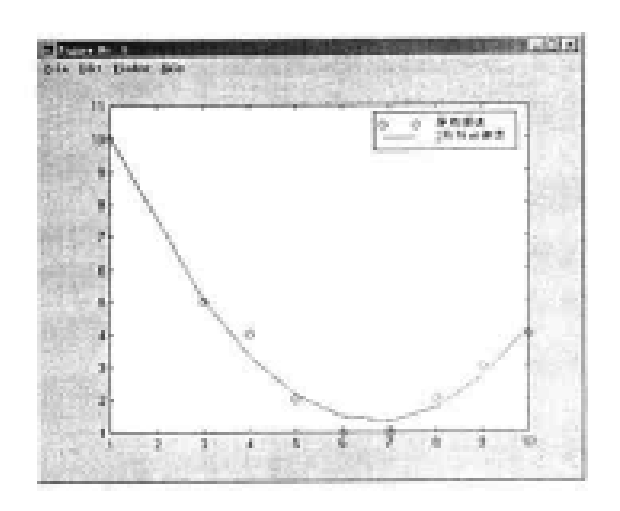


图2.8.2-2　已知数据和拟合曲线图

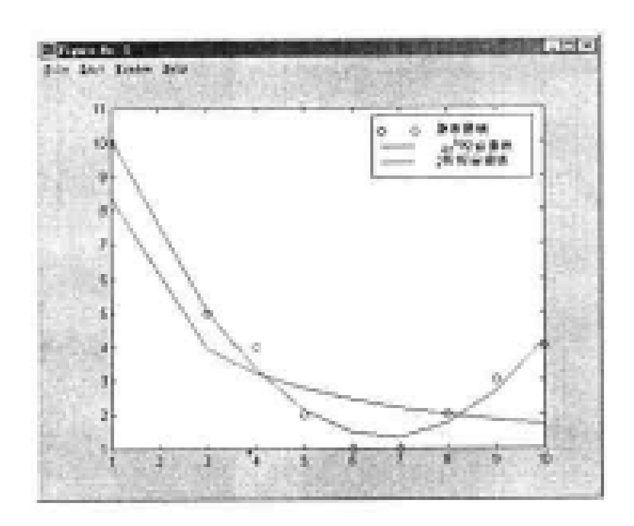


图2.8.2-3　各种拟合曲线比较

2. 用MATLAB中n阶多项式拟合x,y向量给定的数据

函数格式：P=polyfit(x,y,n)

【例 2.8.2-2】 利用【例 2.8.2-1】中(1)的数据拟合 1～6 阶多项式。

```
y1=polyfit(xi,yi,1);
y=poly2str(y1,'x')
f1=polyval(y1,xi);          %按数组运算规则计算多项式值,f1 为多项式,xi 为矩阵
y2=polyfit(xi,yi,2);
y=poly2str(y2,'x')
f2=polyval(y2,xi);          %按数组运算规则计算多项式值
y3=polyfit(xi,yi,3);
y=poly2str(y3,'x')
f3=polyval(y3,xi);          %按数组运算规则计算多项式值
y6=polyfit(xi,yi,6);
y=poly2str(y6,'x')
f6=polyval(y6,xi);          %按数组运算规则计算多项式值
%绘图比较已知数据和拟合曲线图
plot(xi',yi,'bo',xi,f1,'r-',xi,f2,'b-',xi,f3,'m-',xi,f6,'g-')
legend('原数据线','拟合曲线1阶','拟合曲线2阶','拟合曲线3阶','拟合曲线6阶')
```

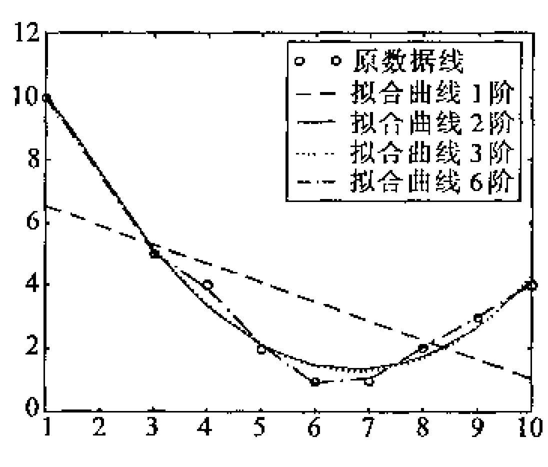


图 2.8.2-4 各种拟合曲线比较

3. 多变量拟合

【例 2.8.2-3】 有已知变量 x_1,x_2,y,如果满足如下关系式 $y=a_0+a_1x_1+a_2x_2$,求系数 a_0,a_1,a_2。

```
x1i=[1 3 4 5 6 7 8 9 10]'
x2i=[0 1 2 3 4 5 6 7 8]'
yi=[10 5 4 2 1 1 2 3 4]'
X=[ones(size(x1i)) x1i x2i]
```

解方程

```
    a -X\yi
a =
      16.6071
     -6.6071
      6.4643
y -16.6071—6.6071x1+6.4643x2
```

2.9　特征值分解

在 2.5.6 节中讲过特征值分解的概念，求 *A* 阵的特征值和特征向量的函数格式如下：

```
d=eig(A)
[V,D]=eig(A)
[V,D]=eig(A,'nobalance')
```

A 是方阵，且它的独立特征向量满秩；d 是 A 阵特征值排成的列向量；D 是 A 阵的特征值对角阵；V 是特征值矩阵，由 A 阵的全部右特征向量构成，且满足 A * V=V * D；当 A 阵中有的元素小到与截断误差相当时，可选用"nobalance"。

【例 2.9-1】　简单实阵的特征值问题。

```
    A=[6 4 3; 4 4 2; 3 2 2];
    d=eig(A)
    [V,D]=eig(A)
d =                    %A 阵特征值排成的列向量
      1
    247/656
    2624/247
V =                    %特征向量矩阵
      881/2158    -1153/2124     491/669
    -881/1079       367/2577     503/899
      881/2158     2022/2443     687/1784
D =                                          %A 阵的特征值对角阵
    1       0          0
    0   247/656        0
    0       0      2624/247
```

2.10　矩阵的分解

MATLAB 解线性方程时常常要用到这样几种分解：

(1) 适用于对称、正定系数矩阵的Cholesky分解。

(2) 系数矩阵为普通方阵的LU分解。

(3) 对于长方阵的正交分解QR分解。

2.10.1 Cholesky分解

Cholesky分解可把矩阵分解为上三角矩阵和其转置的乘积，即 $A=L'L$，其中 L 为上三角矩阵。对于方程 $Ax=b$ 就可变换为 $x=L\backslash(L'\backslash b)$ 形式，以解出方程。

函数格式：L=chol(A)

【例2.10.1-1】 将 A 阵分解为上三角矩阵。

```
    A=pascal(3);
    L=chol(A)
L =
    1    1    1
    0    1    2
    0    0    1
```

可验证 $L' * L=A$。

解方程 $Ax=b$，如果矩阵 A 是对称正定，可先进行Cholesky分解 $L'Lx=b$，即 $x=L\backslash(L'\backslash b)$，使计算更为简单。

2.10.2 三角分解

一个方阵可分解成两个三角阵(上三角阵 U 和下三角阵 L)的乘积，即 $A=L*U$。对于方程 $Ax=b$ 就可变换为 $x=U\backslash(L\backslash b)$ 形式，使计算简化。

命令格式：[L,U]=lu(A)

【例2.10.2-1】 三角分解。

```
    A=[8.1 2.3  -1.5; 0.5  -6.23  0.87;2.5 1.5 10.2]
    [L,U]=lu(A)
L =
    1.0000         0         0
    0.0617    1.0000         0
    0.3086   -0.1240    1.0000
U =
    8.1000    2.3000   -1.5000
         0   -6.3720    0.9626
         0         0   10.7823
%验证
```

```
L*U              %可验证A=L*U
X=inv(A)         %可验证inv(A)=inv(U)*inv(L)
X=inv(U)*inv(L)
D=det(A)         %可验证det(A)=det(L)*det(U)
D=det(L)*det(U)
```

注意:这种算法是基于 Gauss 消去法,而不同于 Doolittle 分解。

```
A=[2   2   3;4   7   7;-2   4   5];
[L,U]=lu(A)
L =
     0.5000   -0.2000    1.0000
     1.0000         0         0
    -0.5000    1.0000         0
U =
    4.0000    7.0000    7.0000
         0    7.5000    8.5000
         0         0    1.2000
%如果用Doolittle分解,则结果为
L0=
   [2 2 3; 0 3 1; 0 0 6]
U0=
   [1 0 0; 2 1 0; -1 2 1]
```

2.10.3　正交分解 (QR 分解)

一个矩阵 *A* 可分解为正交矩阵 *Q* 和上三角阵 *R* 的乘积。

函数格式: [Q,R]=qr(A)

【例 2.10.3-1】 QR 分解及解方程。

```
A=[1   2    2   1;
   2   2/3  -1  2;
   2   1/3  -1  2];
 [Q,R]=qr(A)
Q =
   -1/3    14/15    2/15
   -2/3    -2/15  -11/15
   -2/3    -1/3      2/3
R =
   -3   -4/3   2/3   -3
    0    5/3   7/3    0
```

```
      0         0   1/3      0
%如果解方程 Ax=b，则
    QRx=b
    y=Q\b
    x=R\y
```

2.10.4 矩阵的行列式值、迹、秩、条件数和范数

矩阵的行列式值、迹、秩、条件数和范数，在矩阵分解和解方程中是非常重要的。函数调用格式如下：

det(A)	矩阵 A 的行列式值	Norm(A,1)	矩阵 A 的 1 范数
trace(A)	矩阵 A 迹	Norm(A)	矩阵 A 的 2 范数
rank(A)	矩阵 A 的秩	Norm(A,inf)	矩阵 A 的无穷范数
cond(A)	矩阵 A 的条件数	Norm(A,'fro')	矩阵 A 的 F 范数

【例 2.10.4-1】 验证条件数和迹的定义。

```
    cond=norm(A)*norm(inv(A))
    trace=sum(diag(A))
cond =
    5.3117   %与上题条件数 c 相等
trace=
    3            %与上题迹 t 相等
```

【例 2.10.4-2】 求矩阵的行列式值、迹、秩、条件数和范数。

```
    A=[1 0 1;2 2 1;-1 0 0];
    d=det(A),t=trace(A),rk=rank(A),c=cond(A)
    % 分别是 1 范数、2 范数、无穷范数、f 范数
    n1=norm(A,1),n2=norm(A),ninf=norm(A,inf),nf=norm(A,'fro')
d =
    2
t =
    3
rk =
    3
c =
  5.3117
n1 =
```

```
     4
n2 =
   3.2594
ninf =
     5
nf =
   3.4641
```

2.11 解非线性方程

求解非线性方程 $f(x)=0$，调用函数格式如下：

```
x=fsolve('fname',x0)
x=fsolve('fname',x0,options)
```

第一个输入变量'fname'是已编写好的方程表达式及函数文件名。

options(1) 缺省值为0，若为1，则显示运算中间步骤。

options(2) 缺省值为1e−4，自变量x的最低精度的终止判据。

options(3) 缺省值为1e−4，目标函数f所要求的精度的终止判据。

options(14) 缺省值为500，允许的最大迭代的次数。

【例 2.11-1】 解方程 $f(x)=xe^{x-1}=0$ 的数值解。

解非线性方程，首先需要在编辑器下编一个小程序，然后在命令窗下调用它。步骤是：

(1) 建立函数文件。

用编辑器建立名为nline.m文件以定义方程，此函数文件必须以"function"作为文件头。

```
function q=myx(x)
q=x*exp(x)-1
```

(2) 把写好的文件存放于学生自己的工作目录（例如：\stu）。

(3) 在matlab环境下，利用下述命令使d:\stu成为工作目录。

```
cd d:\stu 或 path(path,'d:\stu')
```

(4) 运行调用积分指令。

在MATLAB命令窗运行以下命令：

```
xx =0.5000          %初值
options(1)=0
xx=fsolve('nline',xx,options)   %options 可省
```

迭代结果：

```
q =
  7.4420e-006
q =
  7.8750e-006
xx =                    %结果
        0.5671
```

【例 2.11-2】 求$\begin{cases}\sin^2 x+y^3+e^z=7\\ 5x^2+3^y-z^3+3=0\\ x-y-z=3\end{cases}$的数值解。

步骤 1：建立函数文件 ff.m。

```
function q=xyz(p)
x=p(1);y=p(2);z=p(3);
q=zeros(3,1);
q(1)=sin(x)^2+y^3+exp(z)-7
q(2)=5*x^2+3^y-z^3+3
q(3)=x-y-z-3
```

步骤 2：在工作窗中运行以下命令。

```
    xyz0=[1,1,1];
    options(18)=0.0000000000001
    xyz=fsolve('ff',xyz0,options)
xyz =
    3.3996    -3.5328    3.9323
```

2.12 数值积分

在数学中，有些被积函数是很难求出它的原函数的，甚至不能求出它的有限形式的原函数。这样就常常用数值积分的方法近似地代替积分计算，往往需要迭代若干次才有比较精确的结果。

在 MATLAB 中，使用quad 和 quad8 求函数数值积分，都采用迭代算法。但是，quad8 比 quad 积分精度更高。

函数格式：

S=quad('fname',a,b,tol,trace) 基于自适应 Simpson 法求数值积分。

S=quad8('fname',a,b,tol,trace) 基于自适应 Newton-Cotes 法求数值积分。

(1) 第一个输入参变量"fname"是被积函数表达式函数文件名。

(2) 第二、第三输入参变量 a 和 b 分别是积分的上下限。

输入参变量 tol 用来控制积分精度。缺省时，tol＝1e－4。

最后输入参变量 trace，若取 1 则用图形展现积分过程，取 0 无图形。缺省时，不画图。

【例 2.12-1】 求 $S=\int_{0.5}^{1}\sqrt{x}\,dx$。

（1）用编辑器建立被积函数 sqrtx.m 文件。

```
function y=sqrtx(x)
y=sqrt(x)
```

（2）把写好的文件存放于学生自己的工作目录（例如：d:\stu）。

（3）利用下述命令使 d:\stu 成为工作目录。

```
cd d:\stu 或 path(path,'d:\stu')
```

（4）运行调用积分指令。

```
S=quad('sqrtx',0.5,1)
S =
    0.4310
```

```
T=trapz(X,Y)          %梯形法求数值积分
O=cumsum(Y)           %欧拉法求数值积分
```

说明：

在 trapz 中，当 X，Y 是同维的列（行）向量时，所得的 T 将是标量，它给出 Y 相对 X 的积分值；当 Y 是（m×n）维矩阵时，X 必须是（m×1）的列向量，积分对 Y 各列分别进行，计算所得（1×n）维 T 的元素是对应列函数对于 X 的积分。在该指令中，X 缺省时，认为 Y 为单位步长等距采样所得的函数阵。

cumsum 的运算结果 O 与 Y 同维。O 的各列给出 Y 相应的积分函数值，且认为积分采用的是等距单位步长（一般地，该积分精度较差）。

【例 2.12-2】 用梯形法和欧拉法求【例 2.12-1】中的数值积分。

```
x=0.5:1/1000:1;nx=length(x);
y=sqrt(x);
T=trapz(y)*1/1000
O=cumsum(y)*1/1000;Ol=O(nx)   %数组 O 的 nx 项，即迭代 nx 次
T=
    0.4310
Ol=
    0.4318
```

当计算数据间隔不等于单位长度的积分时，结果是由间隔增加的倍数去乘 z。

2.13 常微分方程初值问题的数值解

在工程计算中，常微分方程的计算求解是常见的也是很重要的。而一阶的常微分方程的初值问题为：

$$\begin{cases} y'=f(x,y),a\leqslant x\leqslant b \\ y(x_0)=y_0 \end{cases}$$

该函数适用于一阶微分方程。因此，对于高阶微分方程，必须先用替换法化为形如 $\dot{y}=f(x,y)$ 的一阶微分方程组，即"状态方程"。MATLAB 提供的常微分函数均放在\matlab\toolbox\matlab\funfun 下。

非刚性方程的函数

ode23	基于 Runge-Kutta(2,3)法，单步法，低阶，精度较低，速度较快
ode45	基于 Runge-Kutta(4,5)法，单步法，中阶，精度较高，速度较慢
ode113	采用 Adams-Bashforth-Moulton 法，多步法

刚性方程的函数

ode15s	采用 Gearg 法，多步法，精度较低
ode23s	采用二阶改进的 Rosenbrock 法，单步法，速度较快

微分方程的数值解函数的调用格式：[x,y]=odeN('odex',x0,xf,y0,tol,trace)

odeN 可以是 ode23,ode45,ode113,ode15s,ode23s 其中之一。

第一个输入参变量"odex"是定义 $f(x,y)$ 的函数文件名。该函数文件必须以 $\dot{y}=f(x,y)$ 为输出，以 x,y 为输入参变量，次序不能颠倒。

输入参变量 t0 和 tf，分别是积分的初值和终值。

输入参变量 y0 是初始状态列向量。

第五个输入参变量 tol 控制解的精度，缺省值在 ode23 中为 tol=1.e-3；在 ode45 中为 tol=1.e-6。

第六个输入参变量 trace 决定求解的中间结果是否显示，缺省值为 trace=0，不显示中间结果。

【例 2.13-1】 求方程$\begin{cases} y'=y-\dfrac{2x}{y} \\ y(0)=1 \end{cases}$时间区间从 $x=0$ 到 $x=20$ 各节点上的数值解。

(1) 建立函数文件 ode.m。

```
function dy=ode(x,y)
dy=[y-2*x/y]
```

(2) 在命令窗运行：

```
[X,Y]=ode23('ode',[0 20],1)
X=              Y=
   0            1.0e+006 *
   0.0502              1.0000
   ...                 ...
   19.3693             1.1387
   19.6846             1.5278
   20.0000             3.8418
```

(3) 画图观察其变化趋势。

```
plot(X,Y,'-r')
axis([14 20 0 4*10^6])
```

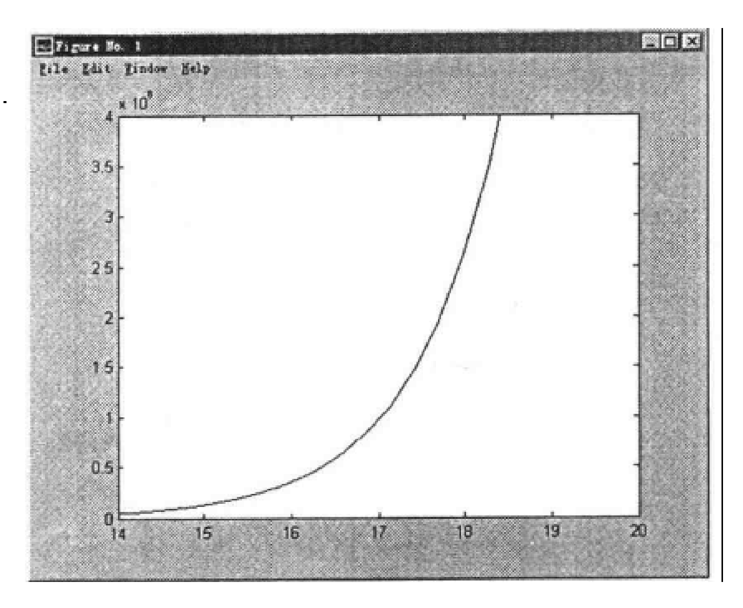


图 2.13-1　方程的图形解

【例 2.13-2】 求方程 $y'''+y'(y''^2-1)-y=0$，初值为 $y(0)=0, y'(0)=1, y''(0)=-1$，时间区间从 $x=0$ 到 $x=1$ 各节点上的数值解。

(1) 将方程化为标准方程。

将微分方程的导数降阶，即令 $y_1=y$， $y_2=y'$， $y_3=y''$，则原方程变为

$$\begin{cases} y_1'=y_2 \\ y_2'=y_3 \\ y_3'=y_2(1-y_3^2)+y_1 \end{cases}$$

其初值条件为 $\begin{cases} y_1(0)=0 \\ y_2(0)=1 \\ y_3(0)=-1 \end{cases}$

(2) 建立函数文件 odex1.m。

```
function ydot=odex1(x,y)
ydot=[y(2);y(3);y(2)*(1-y(3)^2)+y(1)];
```

(3) 解微分方程。

```
[X,Y]=ode45('odex1',[0 20],[0 1 -1]);
```

(4) 画图观察其变化趋势。

```
%plot 为画图函数，参见 5.1.1 节
plot(X,Y(:,1),'-r',X,Y(:,2),':k',X,Y(:,3),'-b')
xlabel('time x');
ylabel('solution Y');
legend('Y1','Y2','Y3')
```

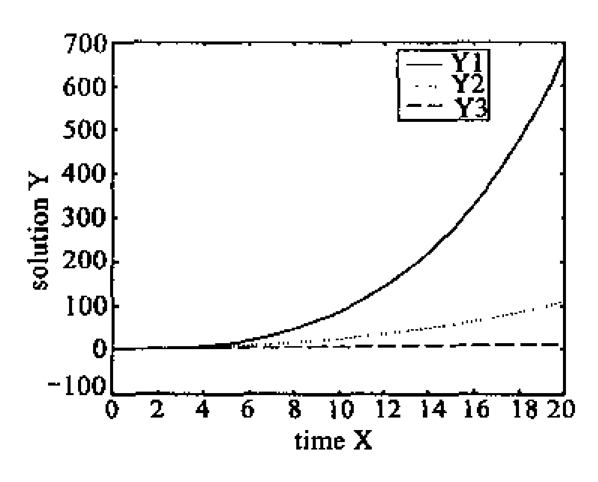


图 2.13-2　方程图形解

习　题　二

1. 用 MATLAB 命令完成矩阵的各种运算。已知矩阵

$$A=\begin{bmatrix}1 & 2 & 3 & 4\\5 & 6 & 7 & 8\\9 & 10 & 11 & 12\\13 & 14 & 15 & 16\end{bmatrix}$$

求出下列的运算结果,并上机验证。

(1) A(:,1)	(2) A(2,:)	(3) A(:,2:3)
(4) A(2:3,2:3)	(5) A(:,1:2:3)	(6) A(2:3)
(7) A(:)	(8) A(:,:)	(9) ones(2,2)
(10) eye(2)	(11) [A,[ones(2,2);eye(2)]]	(12) diag(A)
(13) diag(A,1)	(14) diag(A,-1)	(15) diag(A,2)
(16) tril(A,1)	(17) tril(A,-1)	(18) tril(A,2)
(19) triu(A,1)	(20) triu(A,-1)	(21) triu(A,2)
(22) reshape(A,8,2)	(23) rot90(A)	(24) rot90(A,1)
(25) fliplr(A)	(26) flipud(A)	

2. 用MATLAB命令完成下列矩阵函数运算。

(1) 输入如下矩阵 A:

$$A=\begin{pmatrix}0 & \pi/3\\ \pi/6 & \pi/2\end{pmatrix}$$

(2) 求矩阵 B_1, B_1 中每一元素为对应矩阵 A 中每一元素的正弦函数。

(3) 求矩阵 B_2, B_2 中每一元素为对应矩阵 A 中每一元素的余弦函数。

(4) 求 $B_1^2+B_2^2$。

(5) 求矩阵 A 的特征值与特征矢量,称特征矢量矩阵为 M,而特征值矩阵为 L。

(6) 求 $\mathrm{Msin(L)M^{-1}}$。

(7) 使用funm命令求矩阵 A 的正弦函数,应该得到与(6)相同的结果。

(8) 求 $\cos A$。

(9) 证明 $\sin^2 A+\cos^2 A=I$。

3. 按题目要求用MATLAB命令完成下列矩阵运算。

(1) 使用rand命令产生5个2×2随机矩阵 A,B,C,D,E。

(2) 求下面的矩阵(不使用inv命令):

$$F=A^{-1}[B+C^{-1}(D^{-1}E)]$$

(3) 不使用inv命令,求矩阵 A^{-1}的第1列(仅限使用1个命令)。

使用inv命令验证(2)与(3)的结果。

4. 通过完成下列运算熟悉元素对元素的运算方法及Kronecker乘法。每一部

分先用手算，然后再用MATLAB命令上机验证。现有矩阵为

$$A=\begin{pmatrix}1\\2\\3\end{pmatrix} \qquad B=(4\quad 5\quad 6)$$

(1) kron(A,B') (2) kron(A,B) (3) kron(A',B) (4) kron(A',B')

(5) A. * B' (6) A.\B' (7) A'.\B (8) A'\B

(9) A*inv (B)

5. 首先建立一个随机矩阵 A=fix(10 * (rand(4)))，然后计算下面各题：

(1) exp(A) (2) log(A) (3) sqrt(A) (4) poly(A)

(5) eig(A) (6) roots(A) (7) lu(A) (8) qr(A)

6. 分别用左除和求逆的方法，解下面线性方程：

$$\begin{cases}2x_1+3x_2+5x_3=5\\3x_1+4x_2+7x_3=6\\x_1+3x_2+3x_3=5\end{cases}$$

$$\begin{cases}2x+6y-z=-12\\5x-y+2z=29\\-3x-y+z=5\end{cases}$$

$$\begin{cases}x-y+z=-4\\5x-4y+3z=-12\\2x+y+z=11\end{cases}$$

7. 给定数据：

i	1	2	3	4	5
x_i	4.0	4.2	4.5	4.7	5.1
y_i	102.5	113.2	130.1	142.1	167.5

i	6	7	8	9	10
x_i	5.5	5.9	6.3	6.8	7.1
y_i	195.1	224.8	256.7	299.5	326.7

分别求出形如 $a_0+a_1x+a_2x^2$ 和 ax^b 的最小二乘拟合公式。

8. 计算下面数值积分：

(1) $\sqrt{\frac{2}{\pi}}\int_0^1 e^{-\frac{x^2}{2}}dx$（准确数值为 0.68269）；(2) $\int_0^3 e^x\sin x dx$，要求精确到 10^{-6}；

(3) $\int_0^1 \sqrt{x}dx$，要求精确到 10^{-2}；　　(4) $\int_0^{1.5}\frac{1}{1+x}dx$，要求精确到 10^{-4}。

9. 解下面非线性方程：

(1) $x+\sin(x)-2^x=2$　　(2) $e^x-x+6x=4$

(3) $\begin{cases} x+2y+3z=6 \\ 2x+3\hat{}\ y-5\hat{}\ z+8=0 \\ \cos(x)+y\hat{}\ 3+\lg(z)=8 \end{cases}$

10. 解常微分方程：

(1) 求方程$\begin{cases} y'=1+\frac{2}{y-t} \\ y(0)=2 \end{cases}$ 时间区间从 $t=0$ 到 $t=1$ 各节点上的数值解；

(2) 求方程 $y'''-y''-2*y'=0$，初值为 $y(0)=0, y'(0)=1, y''(0)=-1$，时间区间从 $x=0$ 到 $x=1$ 各节点上的数值解。

第三章 符号计算

在数学运算中，还有很多是符号计算，例如：因式分解、微分积分、求递推公式等等。本章主要有：符号基本计算、极限的运算、级数求和、微分和积分、矩阵的分解、解方程（线性方程、常微分方程）、积分变换，还介绍了符号计算和数值计算之间的关系。

3.1 符号表达式建立

首先，我们必须先生成符号变量、符号表达式，然后才能进行符号计算，否则，MATLAB 认为是误运算。这里要注意的是MATLAB V5.X 和 V4.X 的符号函数有很大的区别，我们讨论的主要是 V5.X 的函数，以后不再赘述。在 MATLAB 中产生符号变量时，不需要特别声明，直接赋给即可，赋值号右边可以是函数、方程、微分方程、矩阵等，但必须加上单引号。

1. 符号表达式

以下定义方法有三种：

(1) 直接赋给：f='x^2+5*x+6'。

(2) 用函数 sym(A)定义：f=sym('x^2+5*x+6')。

其返回结果相同，都是 x^2+5*x+6，但它不是一般的字符串，而是符号表达式，如果式中不写"*"号系统将按出错处理。

(3) 也可用函数 syms 定义符号变量：syms a1 a2 a3，等价于 a1='a1'，a2='a2'，a3='a3'。

2. 符号方程

```
f='x^2+5*x+6=0'      %将方程赋给变量f
df='Dy+x*y=4'        %这里Dy是y'，如果y"写为D2y，y'''为D3y
```

3. 符号矩阵

```
A='[a11 a12;a21 a22]'
B=sym('[a11 a12;a21 a22]')
```

3.2 符号基本计算

3.2.1 符号的四则运算

符号的四则运算均以函数形式给出，符号四则运算的函数如下表：

symadd(A ,B)	给出两个符号矩阵的和 A+B
symsub(A,B)	给出两个符号矩阵的差 A−B
symmul(A,B)	给出乘积 A*B
symdiv(A,B)	实现 A/B，即 AB^{-1}
inv(A)	求 A^{-1}

【例 3.2.1-1】 计算$\frac{1}{(m+1)}$与$\begin{pmatrix} (a+1) & b \\ 0 & (d+3) \end{pmatrix}$相乘。

```
    F=symsub('1/(m+1)',sym('[a+1,b;0,d+3]'))
    G=symmul('1/(m+1)',sym('[a+1,b;0,d+3]'))
F =
    [ 1/(m+1)-a-1,  1/(m+1)-b]
    [     1/(m+1), 1/(m+1)-d-3]
G=
    [1/(s+3),     1/(s+1)/(s+3)*s]
    [      0,1/(s+1)/(s+3)*(s+1.5)]
```

【例 3.2.1-2】 计算$\begin{pmatrix} (a+1) & b \\ 0 & (d+3) \end{pmatrix}$与$\begin{pmatrix} l & m \\ n & p \end{pmatrix}$相除。

```
    A=sym('[a,b;c,d]');
    B=sym('[l,m;n,p]');
    I=sym('[1,0;0,1]')
    C=inv(B)
    C1=symdiv(I,B);              % 与 C 相同
    D=symdiv(A,B)
    D1=symmul(A,inv(B))          % 与 D 相同
C =
    [  p/(l*p-m*n), -m/(l*p-m*n)]
    [-n/(l*p-m*n), l/(l*p-m*n)]
```

```
C1 =
    [  p/(l*p-m*n), -m/(l*p-m*n)]
    [-n/(l*p-m*n),  l/(l*p-m*n)]
D =
    [  (p*a-b*n)/(l*p-m*n), -(-l*b+m*a)/(l*p-m*n)]
    [  (p*c-d*n)/(l*p-m*n), (-m*c+l*d)/(l*p-m*n)]
D1 =
    [  a*p/(l*p-m*n)-b*n/(l*p-m*n), -a*m/(l*p-m*n)+b*l/(l*p-m*n)]
    [  c*p/(l*p-m*n)-d*n/(l*p-m*n), -c*m/(l*p-m*n)+d*l/(l*p-m*n)]
```

3.2.2 符号的幂运算

求幂运算 S^n 的函数格式：sympow(S,n)

S 可以是符号表达式，n 可以是符号或数值表达式；但如果 S 为符号方阵，n 必须为整数。

【例 3.2.2-1】 求符号表达式的幂。

```
A=sym('[a,b;c,d]');
syms a n
A1=sympow(A,2)
a1=sympow(a,2)
a2=sympow(a,n)
A1 =
    [a^2+b*c, a*b+b*d]
    [c*a+d*c, b*c+d^2]
a1 =
    a^2
a2 =
    a^n
```

3.2.3 因式分解

在 MATLAB 中，可以对矩阵的多项式进行分解，化成乘积的形式。其函数格式：

```
factor (X)
```

X 可以是表达式、有理分式、方程、矩阵。

【例 3.2.3-1】 符号矩阵的分解。

```
F=sym('[(x^2+5*x+6),(x^3+3*x^2+3*x+1);...
(x^2+2*x+1),(x^2+7*x+6)]');
```

```
    factor(F)
ans =
    [ (x+3)*(x+2), (x+1)^3]
    [ (x+1)^2, (x+6)*(x+1)]
```

3.2.4 展 开

多项式的展开是因式分解的逆过程。这里是对(包括矩阵元素中的)分子中的因式乘积、三角函数、指数函数、对数函数进行展开,但分母除外。

函数格式:expand(F)

F 可以是矩阵、因式乘积、三角函数、指数函数、对数函数等等。

【例 3.2.4-1】 符号表达式的展开。

```
    FX=sym('[(x+2)*(x+3)/(x+1)^2]');
    expand(FX)
ans =
    1/(x+1)^2*x^2+5/(x+1)^2*x+6/(x+1)^2
```

3.2.5 简 化

简化是指同幂项进行合并。

函数格式:collect(F) 把 F 符号表达式中 x 的同幂项进行合并。

函数格式:collect(F,y) 把 F 符号表达式中 y 的同幂项进行合并。

【例 3.2.5-1】 将多项式中的同幂项进行简化。

```
    syms x
    d=2*x+4*x+6*x^2+3*x^2
    collect(d)
ans =
    6*x+9*x^2
```

【例 3.2.5-2】 将多项式中的同幂项进行简化。

```
    syms x a b y
    F='2*a+3*a*b+4*a*b+7*b';
    Ca=collect(F,a)        %合并"a"的同幂项系数
    Cb=collect(F,b)        %合并"b"的同幂项系数
Ca =
     (2+7*b)*a+7*b
Cb =
     (7*a+7)*b+2*a
```

3.2.6 符号变量替换

符号替换是用数字或字符替换符号表达式中某个变量。MATLAB 中的函数 subs 是利用同类软件 MAPLE 中的 subs 命令编写的，适用于单个符号矩阵、符号表达式、符号代数方程或微分方程，它们的具体使用格式如下：

subs(S,NEW)　　用新变量 NEW 替代 S 中的默认变量。

subs(S,OLD,NEW)　　用新变量 NEW 替代 S 中的指定变量 OLD。

注意：此命令的格式 4.0 与 5.0 版本有很大的区别，OLD 和 NEW 次序截然相反。

【例 3.2.6-1】 符号矩阵的变量替换。

```
syms x a b
F=sym('[cos(a+b)*x,a*x^2+b*x+a;exp(a*x),sqrt(a+x)]');
F1=subs(F,pi)          % 用 pi 替代 F 中默认 x
F2=subs(F,a,7)         % 用 7 替代 F 中的变量 a
F1 =
    [cos(a+b)*pi, a*pi^2+b*pi+a]
    [exp(a*pi), (a+pi)^(1/2)]
F2 =
    [cos(7+b)*x, 7*x^2+b*x+7]
    [exp(7*x), (7+x)^(1/2)]
```

3.2.7 格式转换及计算精度

在 MATLAB 中，允许将数字定义成符号类型，并像数值类型一样设置不同格式。符号的数值表示格式如下表：

格式函数	说　明	4/3 的示例
sym(4/3,'f')	浮点格式	'1.5555555555555'*2^(0)
sym(4/3,'r')	有理格式	4/3
sym(4/3,'e')	有理浮点误差格式	4/3-eps/3
sym(4/3,'d')	十进制格式	1.3333333333333332593184650249896

符号解变成数值解的任意精度：

digits(n)　　使近似解的精度为 n 位有效数字。

vpa(F,n)　　求 F 的 n 位数字有效的近似解，n 缺省时，给出默认精度近似解。

【例 3.2.7-1】　对于 $f(x)=x^2-x-1$，求解 $\frac{d}{dx}f(x)$ 各种表现形式的转换。

```
    syms x
    f='x^2-x-1';
    dx=diff(f);
    Dx=subs(dx,x,'1/3')
    Vdx1=vpa(Dx,20)
    digits(30)
    Vdx=vpa(Dx)
Dx =
    -1/3
Vdx1 =
    -.33333333333333333333
Vdx =
    -.333333333333333333333333333333
```

3.2.8　数值矩阵与符号矩阵的转换

MATLAB 中的数值矩阵不能直接参与符号运算，必须转化为符号表达式才行。将数值矩阵 **n** 转化为符号矩阵的格式：

命令格式：sym(n)

n 为数值矩阵。

【例 3.2.8-1】　将数值矩阵 **n** 转化为符号矩阵。

```
    n=[1/3  sqrt(2)/3; pi  log(2)]
    sym(n,'d')                  %转化为符号矩阵
    Vn=vpa(n,6)
n =
    0.3333    0.4714
    3.1416    0.6931
ans =
[.333333333333333314829616256247,.471404520791031733661924363332]
[3.14159265358979311599796346854,.693147180559945286226763982995]
Vn=
    [.333333,.471403]
    [3.14159,.693147]
```

这样可用第二章学过的命令（如 eye，ones，zeros，magic 等）先生成数值矩阵，再转化成符号阵。

3.3 极　　限

至于极限问题，我们在高等数学中都已经学过。其中有双边极限：$y=\lim\limits_{n\to a}f(n)$；单边极限（左极限、右极限）：$y=\lim\limits_{n\to a-0}f(n)$，$y=\lim\limits_{n\to a+0}f(n)$；无穷极限：$y=\lim\limits_{n\to a}f(n)=\infty$等等。

求极限 $y=\lim\limits_{n\to a}f(n)$的命令格式：limit(f,n,a)

这里 f 是函数的定义，n 为函数的变量，a 为极限变量的趋近点。

【例 3.3-1】 求$\lim\limits_{x\to\infty}(1+\frac{1}{x})^x$；$\lim\limits_{x\to+0}(1+\frac{1}{x})^x$；$\lim\limits_{x\to-1-0}(1+\frac{1}{x})^x$；$\lim\limits_{n\to\infty}(1+\frac{x}{n})^n$。

```
    syms x n
    limit((1+1/x)^ x,x,inf)
    limit((1+1/x)^ x,x,0,'right')
    limit((1+1/x)^ x,x,-1,'left')
    limit((1+x/n)^ n,n,inf)
ans =
    exp(1)
ans =
    1
ans =
    inf
ans =
    exp(x)
```

3.4 级数求和

假设有 $a_1+a_2+\cdots+a_n=\sum\limits_{i=1}^{n}a_n$，称为级数的前 n 项部分和，如果 n 为无穷大，则称为无穷级数。

级数求和函数

symsum(S,v)	关于指定变量 v 对通项表达式 S 求不定和
symsum(S,a,b)	对通项表达式 S 在[a,b]之间求和
symsum(S,v,a,b)	指定变量 v 在[a,b]之间对通项 S 求和

【例 3.4-1】 求 $1+2^2+\cdots+10^2$。

```
    syms k
    symsum(k ^ 2,1,10)
ans=
    385
```

3.5 微分和积分

对于一个函数的微积分的计算，首先是符号运算，然后才能进行数值计算。

符号微积分函数

diff(F,v)	求 F 对变量 v 的一阶导数
diff(F,v,n)	求 F 对变量 v 的 n 阶导数
int(F,v)	求 F 对变量 v 的不定积分
int(F,v,a,b)	求 F 对变量 v 在(a,b)之间内的定积分
taylor(F,v)	求 F 对变量 v 的麦克劳林展开，余项以 6 阶无穷小量给出
taylor(F,v,n)	求 F 对变量 v 的麦克劳林展开，余项以 n 阶无穷小量给出
jacobian(F,v)	求向量 F 的 Jactobian 矩阵，V 是指定变量构成的向量，返回矩阵的 i,j 元素是 dF(i)/dV(j)

【例 3.5-1】 求函数 x^3+3x^2+5x+6 的微分和积分。

```
    f=xx ^ 3+3 * x ^ 2+5x+6';      %定义符号变量
    df=diff(f)                     %对 f 求导
    df2=diff(f,2)                  %对 f 求二阶导数
    intf=int(df)                   %对 df 求积分
    intf2=int(df,0,5)              %对 df 在区间[a,b]求定积分
df=
    3*x ^ 2+6*x+5
df2=
    6*x+6
intf =
    3*x ^ 2+6*x
intf 2=
    105
```

【例 3.5-2】 求$\iint_{\Omega}(x+y)\mathrm{d}x\mathrm{d}y$。

```
    Ix=int('x+y','x')
    syms x y
    Ixy=int(Ix,'y')
Ix =
    1/2*x^2+x*y
Ixy =
    1/2*x^2*y+1/2*x*y^2
```

【例 3.5-3】 求$f(x)=e^x$的麦克劳林展开，最后一项为8阶无穷小量符号。

```
    syms x
    f=exp(x)
    M=taylor(f,8)           %求麦克劳林展开
M=
    1+x+1/2*x^2+1/6*x^3+1/24*x^4+1/120*x^5+1/720*x^6+1/5040
*x^7
```

3.6 矩阵的分解

与数值计算一样，在符号计算中也可以计算矩阵维数、转置、行列式、特征值分解、奇异值分解、零空间和列空间分解等等。其函数格式如下：

[VE,E]=eig(A)	VE给出A的特征向量，E给出矩阵A的特征值。
[VJ,J]=jorda	VJ给出S的广义特征向量，J给出相应的约当标准形。
size(A)	求A矩阵维数。
A'	求A的转置矩阵。
det(sym(A))	求A阵的行列式。
colspace(sym(A))	给出A列空间的基。
NULL(sym(A))	给出A零空间的基。

【例 3.6-1】 求符号矩阵$A=\begin{pmatrix} a & b \\ c & d \end{pmatrix}$的行列式值、逆、特征值。

```
    A=sym('[a,b;c,d]');        %定义矩阵的命令
    DA=determ(A)               %求矩阵A的行列式值
    IA=inv(A)                  %求矩阵A的逆
    EA=eigensys(A)             %求矩阵A的特征值
DA =
    a*d-b*c
```

```
    IA =
    [ d/(a*d-b*c), -b/(a*d-b*c)]
    [ -c/(a*d-b*c), a/(a*d-b*c)]
EA =
    [ 1/2*a+1/2*d+1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
    [ 1/2*a+1/2*d-1/2*(a^2-2*a*d+d^2+4*b*c)^(1/2)]
```

如果有 $\boldsymbol{A}=\begin{pmatrix}1 & 2\\ 3 & 4\end{pmatrix}$，$\boldsymbol{B}=\begin{pmatrix}6\\ 1\end{pmatrix}$

```
    a=[1,2;3,4];b=[6 1]';
    A=sym(a);B=sym(b);           %变成符号阵
    DA=determ(A)                 %求矩阵A的行列式值
    IA=inv(A)                    %求矩阵A的逆
    EA=eigensys(A)               %求矩阵A的特征值
    x=A\B
    X=vpa(x,6)
DA =
    -2
IA =
    [ -2, 1]
    [ 3/2, -1/2]
EA =
    [ 5/2+1/2*33^(1/2)]
    [ 5/2-1/2*33^(1/2)]
x =
    [ -11]
    [ 17/2]
X =
    [    -11.]
    [ 8.50000]
```

【例 3.6-2】 特征值的分解。

```
    D=sym('[1,2;3,4]')
    [VE,E]=eig(D)
    [VEJ,EJ]=jordan(D)
VE =
    [ -1/2+1/6*33^(1/2), -1/2-1/6*33^(1/2)]
    [                 1,                 1]
E =
```

```
    [ 5/2+1/2 * 33^ (1/2),                    0]
    [                    0,  5/2-1/2 * 33^ (1/2)]
VEJ =
    [ -1/22 * 33^ (1/2)+1/2, 1/22 * 33^ (1/2)+1/2]
    [      1/11 * 33^ (1/2),      -1/11 * 33^ (1/2)]
EJ =
    [ 5/2+1/2 * 33^ (1/2),                    0]
    [                    0,  5/2-1/2 * 33^ (1/2)]
```

3.7 解　方　程

3.7.1 符号线性方程的解

与数值解相似，可以求线性方程组 ***A*** * ***X***=***B*** 的解，其格式为：

X=linsolve(A,B) 等价于 X = sym(A)\sym(B)。

【例 3.7.1-1】 解线性方程。

```
    A=sym(magic(3))
    B=sym(fix(10 * randn(3,1)))
    X=linsolve(A,B)
A =
    [ 8, 1, 6]
    [ 3, 5, 7]
    [ 4, 9, 2]
B =
    [  3]
    [  1]
    [ -1]
X =
    [  7/30]
    [ -4/15]
    [  7/30]
```

3.7.2 一般代数方程组的解

对于多元高次方程(组)、特征方程及超越方程(组)，用手工计算非常麻烦，有时是不可思议的事情。MATLAB 使用格式：

solve(F)　　　　对一个符号方程的默认变量求解。

```
solve(F,v)                          对一个符号方程的指定变量 v 求解。
solve(F1,F2,…,FN)                   对 N 个符号方程的默认变量求解。
solve(F1,F2,…,FN,v1,v2,…,vn)       对 N 个符号方程的 v1,v2,…,vn 变量求解。
```

【例 3.7.2-1】 解多元高次方程。

```
syms x y
[x,y]=solve('x^2+3*y+1=0','y^2+4*x+1=0');
x=vpa(x,4)
y=vpa(y,4)
x =
    [ 1.635+3.029*i]
    [ 1.635-3.029*i]
    [         -.283]
    [        -2.987]
y =
    [ 1.834-3.301*i]
    [ 1.834+3.301*i]
    [        -.3600]
    [        -3.307]
```

【例 3.7.2-2】 求矩阵特征方程的根。

```
A=sym(magic(3));                %产生魔方阵
CA=poly(A)                      %求 A 阵的特征多项式 CA
RA=solve(CA);                   %求特征多项式 CA 的根 RA
ra=vpa(RA,4)                    %保留 4 位有效数字
CA =
    x^3-15*x^2-24*x+360
ra =
    [     15.]
    [   4.898]
    [  -4.898]
```

【例 3.7.2-3】 求解超越方程组。

```
[x,y]=solve('log(x/y)=7','x^y=3');
X=vpa(x,4),Y=vpa(y,4)
X =
    223.1
Y =
    .2033
```

3.7.3 常微分方程

在数学中，某个未知的变量和其他变量之间的函数依赖关系是未知的，这些未知的函数关系和某阶的导数连同自变量组成的方程形如：

$$\begin{cases} y' = f(t, y), t_0 \leqslant t \leqslant T & (3.7.3\text{-}1) \\ y(t_0) = y_0 & (3.7.3\text{-}2) \end{cases}$$

式(3.7.3-1)中未知函数为一元的微分方程称为常微分方程。式(3.7.3-2)称为方程的初始条件。

符号常微分方程求解时使用格式为：

$[y1, y2, \cdots] = \text{dsolve}(a_1, a_2, \cdots, a_{12})$

输入参变量由三部分组成：常微分方程、初始条件、自变量，分别用单引号括起来，然后用逗号隔开，其中常微分方程是必不可少的输入内容，其余皆可省略。

常微分方程中，用D表示一阶微分，D_2 表示二阶微分，D_n 表示 n 阶微分，符号 D_2y 表示 d^2y/dt^2。

若要指定自变量，则应位于全部输入参变量 $a1\ a2\cdots$ 中的最后一个。若省略，则默认 x 或 t 为自变量。

初始条件可写成 $y(a)=m$，$Dy(b)=n$ 等。a, b, m, n 可以是变量使用符外的其他字符。当初始条件少于常微分方程数时，在所得解中将出现任意常数符C1，C2…解中任意常数符的数目等于所缺少的初始条件数。

输出变量 $y1, y2, \cdots$ 和应变量数目应该相等以获得结果，也可省略。

【例3.7.3-1】 求 $\frac{dx}{dt} = 3*x + 4*y$，$\frac{dy}{dt} = 5*x - 7*y$ 的通解，及 $x(0)=0$，$y(0)=1$ 的特解。

```
    [x,y]=dsolve('Dx=3*x+4y,Dy=5*x-7*y')
    [x,y]=dsolve('Dx=3*x+4y,Dy=5*x-7*y','x(0)=0,y(0)=1')
x =
    exp(3*t)*C1-4/3                               %C1,C2是任意常数
y =
    1/2*exp(3*t)*C1-1/2*C1*exp(-7*t)+exp(-7*t)*C2-20/21
x =
    exp(3*t)*(-4/3*exp(-3*t)+4/3)
y =
    9/7*exp(-7*t)-20/21+2/3*exp(3*t)
```

【例3.7.3-2】 求解高阶微分方程 $\frac{d^3y}{dt^3} = y + t^2$，初始条件为

$$y(0)=0,\ \frac{\mathrm{d}y(0)}{\mathrm{d}t}=0,\ \frac{\mathrm{d}^2y(0)}{\mathrm{d}t^2}=1.$$

```
y=dsolve('D3y=y+t^2','y(0)=0,Dy(0)=0,D2y(0)=1','t')
y=
-t^2+exp(t)-3^(1/2)*exp(-1/2*t)*sin(1/2*3^(1/2)*t)-exp(-1/2*t)*cos(1/2*3^(1/2)*t)
```

3.7.4 符号函数的图形显示

对于符号函数表达式、微分方程的解，可以用图形显示出来，便于分析。其使用格式如下：

ezplot(f,xmin,xmax)

输入参变量 f 是待绘的符号函数。xmin,xmax 是绘图的自变量 x 的范围，可以缺省，缺省值为[-2π,2π]。

【例 3.7.4-1】绘制【例 3.7.3-2】微分方程所得的解 $y(t)$ 的图形（图 3.7.4-1）。

```
ezplot(y,-6,6)          %绘制微分方程的解 y(t)在[-6,6]中的图形
```

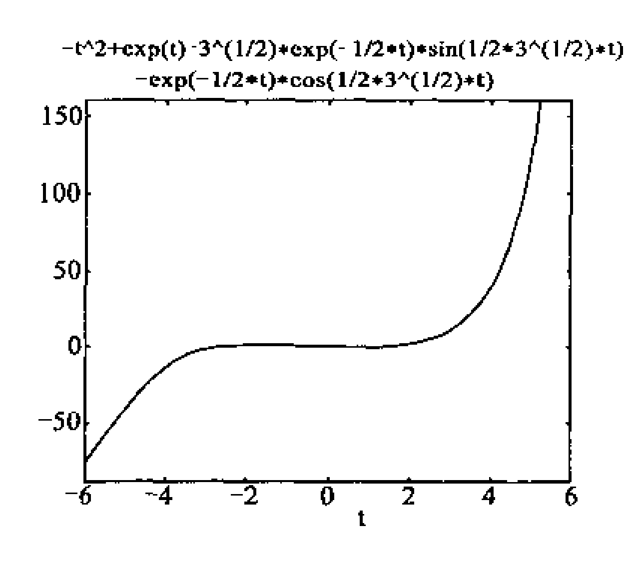


图 3.7.4-1 微分方程的图形解

3.8 积分变换

积分变换是机类、电类及从事控制、通讯等专业的数学基础，本节将介绍 MATLAB 关于积分变换的用法。

3.8.1 傅里叶Fourier积分变换

1. 傅里叶Fourier积分变换及逆变换的概念

若有函数$f(t)$成立

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}\left[\int_{-\infty}^{+\infty}f(\tau)e^{-j\omega\tau}d\tau\right]e^{j\omega t}d\omega \tag{3.8.1-1}$$

则有

$$F(\omega)=\int_{-\infty}^{+\infty}f(t)e^{-j\omega t}dt \tag{3.8.1-2}$$

用MATLAB表示:F(w) = int(f(t) * exp(—i * w * t),t,—inf,inf)

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{+\infty}F(\omega)e^{j\omega t}d\omega \tag{3.8.1-3}$$

式(3.8.1-2)称作傅氏变换式;式(3.8.1-3)称作傅氏逆变换式;$F(\omega)$称作$f(t)$的相函数。

2. 傅里叶变换函数

F = FOURIER(f) 关于缺省独立变量x的字符向量f的傅里叶变换,缺省返回为w的函数,即F(w) = int(f(t) * exp(—j * w * t),t,—inf,inf)。

F = FOURIER(f,v) 缺省独立变量w关于字符向量v的傅里叶变换,即F(v) = int(f(x) * exp(—j * v * x),x,—inf,inf)。

FOURIER(f,u,v) 缺省独立变量x的关于字符向量u的傅里叶变换,即F(v) = int(f(u) * exp(—j * v * u),u,—inf,inf)。

【例3.8.1-1】 计算各种傅里叶变换。

```
    syms t v w x
    fourier(1/t)
    fourier(exp(-x^2),x,t)
    fourier(exp(-t)*sym('Heaviside(t)'),v)
    fourier(diff(sym('F(x)')),x,w)
ans= i*pi*(Heaviside(-w)-Heaviside(w))
ans= pi^(1/2)*exp(-1/4*t^2)
ans= 1/(1+i*v)
ans= i*w*fourier(F(x),x,w)
```

3. 傅里叶逆变换函数

f = IFOURIER(F) 是缺省独立变量w的关于符号向量F的傅里叶逆变换,缺省返回为关于x的函数:f(x) = 1/(2 * pi) * int(F(w) * exp(i * w * x),w,

—inf,inf)。

f = IFOURIER(F,u) 进行一个关于 u 代替缺省 x 项的傅里叶逆变换,即 f(u) = 1/(2 * pi) * int(F(w) * exp(i * w * u,w,—inf,inf)。

f = IFOURIER(F,v,u) 进行一个关于 v 代替缺省 w 项的傅里叶逆变换,即 f(u) = 1/(2 * pi) * int(F(v) * exp(i * v * u,v,—inf,inf)。

【例 3.8.1-2】 计算各种傅里叶逆变换。

```
    syms t u w x v
    ifourier(w * exp(-3 * w) * sym('Heaviside(w)'))
    ifourier(1/(1+w^ 2),u)
    ifourier(v/(1 + w^ 2),v,u)
    ifourier(sym('fourier(f(x),x,w)'),w,x)
ans=
    1/2/pi/(3-i * t)^ 2
ans=
    1/2 * exp(-u) * Heaviside(u)+1/2 * exp(u) * Heaviside(-u)
ans=
    i/(1+w^ 2) * Dirac(1,-u)
ans=
    f(x)
```

4. 快速傅里叶变换

(1) 快速傅里叶变换概念

快速傅里叶变换就是关于矢量 X 的离散傅里叶变换(DFT)快速算法,对于一个长度为 N 的有限长序列 $x[n]$,它的 DFT 算法为

$$X(k)=\sum_{n=0}^{N-1}x[n]W_N^{kn},\ (k=0,1,2,\cdots,N-1) \qquad (3.8.1\text{-}4)$$

逆变换:

$$x[n]=\frac{1}{N}\sum_{k=0}^{N-1}X(k)W_N^{-kn},(n=0,1,2,\cdots,N-1) \qquad (3.8.1\text{-}5)$$

对于有限数列 $x[n]$的长度 N 很大的情况下,直接计算 DFT 要花费很长时间,那么按时间或按频率抽取一些子序列来计算,则要快得多。

如果 X 向量长度 N 是 2 的整数次方的形式,则可使用基数 2 快速傅里叶变换算法,否则只能使用一个较慢的混合基算法。

对于矩阵,FFT 算法是按列进行计算的。

(2) 快速傅里叶变换函数

FFT(X)是关于 X 的快速傅里叶变换。

FFT(X,N) 是 N 点 FFT 算法，如果少于 N 点，则用零填满；如果多于 N 点，则舍弃多余的部分。

FFT(X,[ ],DIM) 或 FFT(X,N,DIM)用于跨维 DIM 的操作。

有关函数还有：FFT2，IFFT2，FFTSHIFT。

(3) 快速傅里叶逆变换函数

IFFT(X) 是关于 X 的快速傅里叶逆变换。

IFFT(X,N) 是 N 点快速傅里叶逆变换。

IFFT(X,[ ],DIM) or IFFT(X,N,DIM) 用于跨维 DIM 的关于 X 的离散傅里叶逆变换。

3.8.2　拉普拉斯 Laplace 变换

1. 拉普拉斯变换概念

$$L(s) = \int_0^{+\infty} f(t)e^{-st}dt \tag{3.8.2-1}$$

记为：$F(s)=L[f(t)]$，$F(s)$称为$f(t)$的拉氏变换，而$f(t)$称为$F(s)$的拉氏逆变换，记为：

$f(t)=L^{-1}[F(s)]$

2. 拉普拉斯变换函数

L = LAPLACE(F) 是缺省独立变量 t 的关于符号向量 F 的拉氏变换，缺省返回为关于 s 的函数，即

L(s) = int(F(t) * exp(-s * t),0,inf)

L = LAPLACE(F,t) 是一个关于 t 而代替缺省 s 项的拉氏变换，即

L(t) = int(F(x) * exp(-t * x),0,inf)

L = LAPLACE(F,w,z) 是一个关于 z 而代替缺省 s 项的拉氏变换，即

L(z) = int(F(w) * exp(-z * w),0,inf)

【例 3.8.2-1】 计算各种拉普拉斯变换。

```
syms a s t w x
    laplace(t^5)
ans= 120/s^6
    laplace(exp(a * s))
ans= 1/(t-a)
    laplace(sin(w * x),t)
ans= w/(t^2+w^2)
    laplace(cos(x * w),w,t)
```

```
ans = t/(t^2+x^2)
     laplace(x^sym(3/2),t)
ans = 3/4*pi^(1/2)/t^(5/2)
     laplace(diff(sym('F(t)')))
ans = laplace(F(t),t,s)*s-F(0)
```

3. 拉普拉斯逆变换函数

F = ILAPLACE(L) 是缺省独立变量 s 的关于符号向量 L 的拉氏逆变换，缺省返回是关于 t 的函数 F(t) = int(L(s) * exp(s * t),s,c-i * inf,c+i * inf)。

ILAPLACE(L,y) 是一个关于 y 而代替缺省 t 项的拉氏逆变换，即

F(y) = int(L(y) * exp(s * y),s,c-i * inf,c+i * inf)

ILAPLACE(L,y,x) 是一个关于 x 而代替缺省 t 项的拉氏逆变换，即

F(y) = int(L(y) * exp(x * y),y,c-i * inf,c+i * inf)

【例 3.8.2-2】 计算各种拉普拉斯逆变换。

```
     syms s t w x y
     ilaplace(1/(s-1))
ans = exp(t)
     ilaplace(1/(t^2+1))
ans = sin(x)
     ilaplace(t^(-sym(5/2)),x)
ans = 4/3/pi^(1/2)*x^(3/2)
     ilaplace(y/(y^2+w^2),y,x)
ans = cos(w*x)
     ilaplace(sym('laplace(F(x),x,s)'),s,x)
ans = F(x)
```

3.8.3 Z 变换

1. Z 变换和 Z 逆变换概念

上面讨论的拉氏变换可视为连续时间傅里叶变换的推广。在连续时间 LTI 系统的分析中，拉氏变换是一个强有力的工具。在离散时间情况下，与拉氏变换相对应的是 Z 变换。Z 变换实际上是离散时间函数的拉氏变换。

设连续信号 $f(t)$ 的拉氏变换为 $F(s)$，$f(t)$ 经采样开关后变为离散时间函数 $f*(t)$，采样周期为 T，设 $T<0$ 时，$f(t)=0$，则

$$f*(t) = \sum_{k=0}^{+\infty} f(kT)\delta(t-kT) \tag{3.8.3-1}$$

对式(3.8.3-1)进行拉氏变换得

$$F*(s)=\sum_{k=0}^{+\infty}f(kT)e^{-skT} \tag{3.8.3-2}$$

采用信号的拉氏变换是 s 的超越函数，为了方便，引进新变量，令

$$z=e^{sT}$$

将 $F*(s)$的函数变为 z 的函数，则采样信号的 z 变换：

$$F*(z)=\sum_{k=0}^{+\infty}f(kT)e^{-k} \tag{3.8.3-3}$$

若式(3.8.3-3)的级数收敛，则称 $F(z)$为 $f*(t)$的 z 变换。记作 $F(z)=Z[f*(t)]$。

z 变换与傅氏变换之间也存在重要的关系。将复变量 z 表示成极坐标形式：

$$z=re^{j\Omega}$$

用 r 表示 z 的模，Ω 表示它的相位，则 z 变换可表示为

$$F*(z)_{z=e^{j\Omega}}=F(e^{j\Omega}) \tag{3.8.3-4}$$

2. Z 变换函数

F = ZTRANS(f) 是缺省独立变量 n 的关于符号向量 f 的 Z 变换，缺省返回是关于 z 的函数：

F(z) = symsum(f(n)/z^ n, n, 0, inf)

ZTRANS(f,w) 是一个关于 w 而代替缺省 z 项的 Z 变换：

F(w) = symsum(f(n)/w^ n, n, 0, inf)

ZTRANS(f,k,w) 将 f 变成为符号变量 k 的函数：

F(w) = symsum(f(k)/w^ k, k, 0, inf)

【例 3.8.3-1】 计算各种 Z 变换。

```
    syms k n w z
    ztrans(2^ n)
ans= z/(z-2)
    ztrans(sin(k*n),w)
ans= sin(k)*w/(1-2*w*cos(k)+w^ 2)
    ztrans(cos(n*k),k,z)
ans= z*(-cos(n)+z)/(-2*z*cos(n)+z^ 2+1)
    ztrans(cos(n*k),n,w)
ans= w*(-cos(k)+w)/(-2*w*cos(k)+w^ 2+1)
    ztrans(sym('f(n+1)'))
ans= z*ztrans(f(n),n,z)-f(0)*z
```

3. Z 逆变换函数

f = IZTRANS(F) 是缺省独立变量 z 的关于符号向量 F 的 Z 变换，缺省返回

是关于 n 的函数。

f = IZTRANS(F,k)将 f 变成一个关于 k 而代替缺省 n 的 Z 的函数。

f = IZTRANS(F,w,k) 将 F 变为省缺项为 symvar(F)的关于 w 的函数,缺省返回是关于 k 的函数。

【例 3.8.3-2】 计算各种 Z 逆变换。

```
    syms z k
    iztrans(z/(z—2))
ans= 2^n
    iztrans(exp(x/z),z,k)
ans= x^k/k!
```

习 题 三

1. 对下列函数求导:

(1) x^n　(2) $\frac{1}{x}$　(3) $\frac{1}{x^n}$　(4) $\sqrt[n]{x}$

(5) e^x　(6) a^x　(7) x^x　(8) $\ln x$

(9) $\log_a x$　(10) $\lg x$　(11) $\sin x$　(12) $\sin(x^3+y^4)$

(13) $\ln(x+\sqrt{1+x^2})$　(14) $\ln(x+\sqrt{x^2-1})$　(15) $\frac{1}{2}\ln\frac{1+x}{1-x}$

2. 对下列函数求积分:

(1) $\int a^x\mathrm{d}x$　(2) $\int\tan x\mathrm{d}x$　(3) $\int\sin^2 x\mathrm{d}x$　(4) $\int_0^{2\pi}\cos^2 x\mathrm{d}x$

(5) $\int\frac{1}{a^2-x^2}\mathrm{d}x$　(6) $\int_0^a\frac{1}{a^2+x^2}\mathrm{d}x$　(7) $\int_0^{a/2}\frac{1}{\sqrt{a^2-x^2}}\mathrm{d}x$　(8) $\int_0^{a/2}\frac{1}{\sqrt{a^2+x^2}}\mathrm{d}x$

3. 对下列函数求极限:

(1) $\lim\limits_{x\to 0}\frac{\sin x}{x}$　(2) $\lim\limits_{x\to\infty}(\frac{x}{1+x})^x$　(3) $\lim\limits_{x\to\infty}(1+\frac{2}{x})^{3x}$　(4) $\lim\limits_{x\to\infty}(1+\frac{1}{n}+\frac{1}{n^2})^n$

4. 对下列函数麦克劳林展开,最后一项为 8 阶无穷小量。

(1) $\sin x$　(2) $\cos^3 x$　(3) $\ln(x+\sqrt{1+x^2})$

(4) $(1+x)^m$　(5) $e^{-x^2}\sin x$　(6) $\frac{1}{1-x-x^2}$

5. 求微分方程,并画出图形解。

(1) 求 $\frac{\mathrm{d}x}{\mathrm{d}t}=2x-3y,\frac{\mathrm{d}y}{\mathrm{d}t}=5x+7y$ 的通解,及满足 $x(0)=0,y(0)=1$ 的特解。

(2) 求解高阶微分方程 $\frac{\mathrm{d}^3y}{\mathrm{d}t^3}=\frac{\mathrm{d}^2y}{\mathrm{d}t^2}-\frac{\mathrm{d}y}{\mathrm{d}t}-y+t^2$,初始条件为 $y(0)=0$,

$\frac{dy(0)}{dt}=1, \frac{d^2y(0)}{dt^2}=-1$。

(3) 解方程组 $\begin{cases}\frac{df}{dt}=3f+4g \\ \frac{dg}{dt}=-5f-7g\end{cases}$ 的通解，及满足 $f(0)=0$，$g(0)=1$ 的特解。

第四章　数据分析和统计

统计学是关于数据处理的一门科学。它可通过收集来的数据，从中分析、归纳出事物所蕴含的本质，以便发现新知识。在新的世纪中，随着经济和科学事业的发展，人们需要处理的数据越来越多，统计学应用的领域越来越广泛，这样也就需要我们更加重视统计学。

MATLAB 提供了一套数值计算环境的统计分析工具，具有很强的工程和科学统计的计算能力，其中包括 200 多个函数或命令。本章将分为概率分布、参数估计、描述性统计、方差及回归分析、假设检验、统计绘图、统计工序管理等不同内容来介绍。

4.1　概 率 分 布

MATLAB 统计工具箱提供了 20 多个概率分布函数，可分四个部分。

4.1.1　概率密度函数

对随机变量的统计取决于其概率的分布。随机变量的概率分布主要可分为离散型、连续型两大类。离散型随机变量可能取到的值是有限个或可列无限多个，其概率密度函数是观察到的特定值的概率；连续型随机变量如果存在非负函数 $p(x)\geqslant 0$，使对于任意实数 $a\leqslant b$ 有 $P\{a<X<b\}=\int_a^b p(x)\,\mathrm{d}x$，则函数 $p(x)$ 称为 X 的概率密度函数，它满足在整个区间积分为 1，而对观察到的某个特定值的概率为 0。在 MATLAB 中，这种概率密度函数常常表示为 pdf 函数。它并不是单一的函数，而是以一个或多个为特征的函数族。一旦有了参数，其函数才惟一确定。有关特定分布的 pdf 函数和通用分布函数均在表 4-1 和表 4-2 中列出。

表 4-1 **常用离散型分布**

分布类型	概率密度函数数学表达式	命令格式	说 明
通用分布函数		Y=pdf('name', X,A1,A2,A3)	'name'为特定分布类型名；X为自变量取值矩阵；A1,A2,A3为分布参数
二项分布 $B(n,p)$	$y=\binom{n}{x}p^x q^{n-x}$；$p>0$，$q>0$，$x=0,1,\cdots,n$；$p+q=1$，$n>0$	Y=binopdf(X,N,P)	N为正整数，P在[0,1]区间内取数
泊松分布 $P(\lambda)$	$y=\frac{\lambda^x}{x!}e^{-\lambda}$；$x=0,1,2,\cdots$；$\lambda$为正实数	Y = poisspdf (X, LAMBDA)	参数LAMBDA必须为正数;X为任何非负整数,否则Y为0
几何分布 $G(p)$	$y=pq^{x-1}$；$x=0,1,2,\cdots$ $p>0$，$q>0$；$p+q=1$	Y=geopdf(X,P)	参数P在[0,1]区间内取数
负二项分布 $B^-(r,p)$	$y=\binom{r+x+1}{x}p^x q^r$ $x=0,1,2,\cdots$；$r>0$，$p>0$，$q>0$；$p+q=1$	Y=nbinpdf(X,R,P)	
超几何分布 $H(M,K,n)$	$y=\frac{\binom{K}{x}\binom{M-K}{n-x}}{\binom{M}{n}}$	Y=hygepdf(X,M,K,N)	参数M,K,N为正整数；参数X小于或等于其他参数值；参数N小于或等于M．
离散均匀分布 $P(n)$	$y=\frac{x}{N}$；$x=1\cdots N$	Y=unidpdf(X,N)	参数N为正整数

表 4-2 **常用连续型分布**

分布类型	概率密度函数数学表达式	命令格式	命令调用说明
连续均匀分布 $u(a,b)$	$y(x)=\begin{cases}\frac{1}{b-a}, & a\leqslant x\leqslant b\\ 0, & x<a \text{ 或 } x>b\end{cases}$	Y = unifpdf (X,A,B)	A<B
正态分布 $N(\mu,\sigma)$	$y(x)=\frac{1}{\sigma\sqrt{2\pi}}e^{\frac{-(x-\mu)^2}{2\sigma^2}}$	Y = normpdf (X, MU, SIGMA)	参数SIGMA为正
瑞利分布 $R(\mu)$	$y(x)=\begin{cases}\frac{x}{b^2}e^{(\frac{-x^2}{2b^2})}, & x\geqslant 0; b>0,\\ 0, & x<0\end{cases}$	Y = raylpdf (X,B)	
指数分布 $e(\mu)$	$y(x)=\frac{1}{\mu}e^{-\frac{x}{\mu}}$	Y = expppdf (X,MU)	参数MU为正

续表

分布类型	概率密度函数数学表达式	命令格式	命令调用说明
β 分布 $\beta(a,b)$	$y(x)=\begin{cases}\frac{x^{a-1}(1-x)^{b-1}}{B(a,b)},0<x<1, \\ 0,x\leqslant 0 \text{ 或 } x\geqslant 1,\end{cases}$ $a>0,b>0$	Y = betapdf (X,A,B)	-
γ 分布 $\Gamma(a,b)$	$y(x)=\frac{1}{b^a\Gamma(a)}x^{a-1}e^{\frac{x}{b}}$	Y = gampdf (X,A,B)	参数 A 和 B 为正整数，X 在区间[0,∞]内取值
对数正态分布 $Ln(\mu,\sigma^2)$	$y(x)=\begin{cases}\frac{1}{x\sigma\sqrt{2\pi}}e^{\frac{-(\ln x-\mu)^2}{2\sigma^2}}, \\ x>0, \\ 0,x\leqslant 0\end{cases}$	Y = lognpdf (X,MU, SIGMA)	
χ^2 分布 $\chi^2(v)$	$y(x)=\begin{cases}\frac{x^{(v-2)/2}e^{-x/2}}{2^{v/2}\Gamma(v/2)},x>0, \\ 0,x\leqslant 0\end{cases}$	Y=chi2pdf (X,V)	V 为正整数
F 分布 $F(v_1,v_2)$	$y(x)=\frac{\Gamma\left[\frac{v_1+v_2}{2}\right]}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_2}{2}\right)}\left(\frac{v_1}{v_2}\right)^{\frac{v_1}{2}}\frac{x^{\frac{v_1-2}{2}}}{\left[1+\left(\frac{v_1}{v_2}\right)x\right]^{\frac{v_1+v_2}{2}}}$	Y = fpdf (X, V1,V2)	参数 V1 和 V2 为正整数，X 在区间[0,∞]内取值
t 分布 $t(n)$	$y(x)=\frac{\Gamma(\frac{v+1}{2})}{\sqrt{v\pi}\Gamma(\frac{v}{2})}\frac{1}{(1+\frac{x^2}{v})^{\frac{v+1}{2}}}$	Y = tpdf (X, V)	V 为正整数
威布尔分布 $W(a,b)$	$y(x)=abx^{b-1}e^{-ax^b}$	Y = weibpdf (X,A,B)	参数 A,B 为整数

【例 4.1.1-1】 计算正态分布。

```
x = [-2:0.5:2];
f = normpdf(x,0,1);
fpdf =pdf('norm',x,0,1); %结果与 f 相同
%x 为计算概率密度的点集,0 是位置参数(均值),1 是散度参数(标准差)
f =
  Columns 1 through 7
    0.0540   0.1295   0.2420   0.3521   0.3989   0.3521   0.2420
```

```
Columns 8 through 9
  0.1295   0.0540
```

4.1.2 累积分布函数与逆累积分布函数

连续型随机变量的累积分布函数(cdf)称为分布函数，完全取决于其概率密度$p(x)$，则有

$$F(x)=\int_{-\infty}^{x}p(u)\mathrm{d}u$$

若f是概率密度函数，则相应的累积分布函数F为

$$F(x)=P(X\leqslant x)=\int_{-\infty}^{x}f(t)\mathrm{d}t \qquad (4.1.2\text{-}1)$$

(4.1.2-1)式累积分布函数$F(x)$表示所观察结果小于或等于的概率有两个性质：

- cdf 值$F(x)$的范围是1到0；
- 若$y\geqslant x$，则$F(x)\geqslant F(x)$。

逆累积分布函数(icdf 或 inv)返回预定概率条件下的假设检验的临界值，实质上是cdf 的逆函数。有关特定分布的cdf 和icdf 函数和通用分布函数均在表4-3和表4-4中列出。

表4-3 常用离散型分布

分布类型	概率密度函数数学表达式	Cdf 函数格式	Inv 函数格式
通用分布函数		Y = cdf('name', X, A1, A2, A)	Y = icdf('name', P, A1, A2, A)
二项分布 $B(n,p)$	$y=\sum_{i=0}^{x}\binom{n}{i}p^iq^{n-i}$；$p>0$，$q>0$，$x=0,1,\cdots,n$；$p+q=1$；$n>0$	Y = binocdf (X, N, P)	X=binoinv(Y, N, P)
泊松分布 $P(\lambda)$	$y=\mathrm{e}^{-\lambda}\sum_{i=0}^{floor(x)}\frac{\lambda^i}{i!}$ $x=0,1,2,\cdots$；λ为正实数	P = poisscdf (X, LAMBDA)	X=poissinv(P, LAMBDA)
几何分布 $G(p)$	$y=\sum_{i=0}^{floor(x)}pq$；$x=0,1,2,\cdots$；$p>0,q>0$；$p+q=1$	Y=geocdf(X,P)	X=geoinv(Y,P)

续表

分布类型	概率密度函数数学表达式	Cdf 函数格式	Inv 函数格式
负二项分布 $B^-(r,p)$	$y=\sum_{i=0}^{x}\binom{r+i+1}{i}p^r q^i$, $x=0,1,2,\cdots;r>0$, $p>0,q>0;p+q=1$	Y = nbincdf (X, R, P)	X=nbininv(Y,R,P)
超几何分布 $H(M,K,n)$	$y=\sum_{i=0}^{x}\frac{\binom{K}{i}\binom{M-K}{N-i}}{\binom{M}{N}}$	P = hygecdf (X, M, K,N)	X=hygeinv(P,M,K,N)
离散均匀分布 $P(n)$	$y=\frac{floor(x)}{N};x=1\cdots N$	P=unidcdf(X,N)	X=unidinv(P,N)

表 4-4　**常用连续型分布**

分布类型	概率密度函数数学表达式	Cdf 函数格式	Inv 函数格式
连续分布 $u(a,b)$	$y(x)=\begin{cases}\frac{x-a}{b-a}, a\leqslant x\leqslant b\\ 0, x<a \text{ 或 } x>b\end{cases}$	P=unifcdf(X,A,B)	X=unifinv(P,N)
正态分布 $N(\mu,\sigma)$	$y(x)=\frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{x}e^{\frac{-(t-\mu)^2}{2\sigma^2}}dt$	Y = normcdf (X, MU,SIGMA)	X=norminv(P, MU, SIGMA)
瑞利分布 $R(\mu)$	$y(x)=\int_0^x\frac{t}{b^2}e^{(\frac{-t^2}{2b^2})}dt$	P=raylcdf(X,B)	X=raylinv(P,B)
指数分布 $e(\mu)$	$y(x)=\int_0^x\frac{1}{\mu}e^{-\frac{x}{\mu}}dt=1-e^{-\frac{x}{\mu}}$	P=exppcdf(X, MU)	X=exppinv (P,MU)
β分布 $\beta(a,b)$	$y(x)=\frac{1}{B(a,b)\int_0^x t^{a-1}(1-t)^{b-1}dt}$	P = betacdf (X, A,B)	X = betainv (P, A, B)
γ分布 $\Gamma(a,b)$	$y(x)=\frac{1}{b^a\Gamma(a)}\int_0^x t^{a-1}e^{\frac{t}{b}}dt$	P=gamcdf(X,A,B)	X = gaminv (P, A, B)

续表

分布类型	概率密度函数数学表达式	Cdf 函数格式	Inv 函数格式
对数正态分布 $Ln(\mu,\sigma^2)$	$y(x)=\frac{1}{x\sigma\sqrt{2\pi}}\int_0^x \frac{e^{-\frac{(\ln t-\mu)^2}{2\sigma^2}}}{t}dt$	P=logncdf(X, MU,SIGMA)	X=logninv(P, MU, SIGMA)
χ^2 分布 $\chi^2(v)$	$y(x)=\int_0^x \frac{t^{(v-2)/2}e^{-t/2}}{2^{v/2}\Gamma(v/2)}dt$	P=chi2cdf(X,V)	X=chi2inv(P, V)
F 分布自由度 (v_1,v_2) $F(v_1,v_2)$	$y(x)=\int_0^x \frac{\Gamma\left[\frac{v_1+v_2}{2}\right]}{\Gamma\left(\frac{v_1}{2}\right)\Gamma\left(\frac{v_2}{2}\right)}\left(\frac{v_1}{v_2}\right)^{\frac{v_1}{2}}\frac{t^{\frac{v_1-2}{2}}}{\left[1+\left(\frac{v_1}{v_2}\right)t\right]^{\frac{v_1+v_2}{2}}}dt$	P = fcdf (X, V1, V2)	X = finv (P, V1,V2)
t 分布 $t(n)$	$y(x)=\int_{-\infty}^x \frac{\Gamma(\frac{v+1}{2})}{\sqrt{v\pi}\Gamma(\frac{v}{2})(1+\frac{t^2}{v})^{\frac{v+1}{2}}}dt$	P=tpdf(X,V)	X=tinv(P,V)
威布尔分布 $W(a,b)$	$y(x)=\int_0^x abt^{b-1}e^{-at^b}=1-e^{-ax^b}$	P = weibcdf (X, A,B)	X=weibinv(P, A,B)

【例 4.1-2】 用正态分布说明 cdf 和 inv 函数之间的关系。

```
    x = [-1:0.5:1]
    xnew = norminv(normcdf(x,0,1),0,1)
x =
    -1.0000    -0.5000        0    0.5000    1.0000
xnew =
    -1.0000    -0.5000        0    0.5000    1.0000
```

说明:连续分布中取值点的 cdf 计算值为 0 至 1 的概率值,这些概率值的逆 icdf 则给出其原来的取值点。

【例 4.1.2-2】 算出正态分布的 80%置信区间。

```
    p=[0.013 0.813]
    x=norminv(p,0,1)
x =
    -2.2262    0.8890
```

说明:变量 x 中的值在给定概率区间 p 的条件下,有参数 0 和 1 所确定的正态分布函数的逆函数结果,p(2)-p(1)=0.8,所以 x 给出了正态分布 80%的置信

区间。

4.1.3 随机数生成函数

所有分布的随机数的产生都是开始于均匀分布随机数。对于 20 种分布类型，每种分布的随机数都可任意产生。函数可产生单个随机数或随机数矩阵，这取决于函数调用时所用的参数情况。MATLAB 所有的特定分布的随机产生函数和通用分布的相关函数都列在表 4-5 和表 4-6 内。至于命令中的参变量含义，请参阅相应分布的数学表达式。

表 4-5 常用离散型分布

分布类型	函数调用格式		
通用分布函数	R＝random('name',A1,A2,A3,m,n)		'name'为分布类型名；A1,A2,A3 为分布参数；m,n 为行列维。
二项分布	R＝binornd(N,P)	R＝binornd(N,P,mm)	R＝binornd(N,P,mm,nn)
泊松分布	R ＝poissrnd (LAMBDA)	R ＝poissrnd (LAMBDA,m)	R＝poissrnd (LAMBDA,m,n)
几何分布	R＝geornd(P)	R＝geornd(P,m)	R＝geornd(P,m,n)
负二项分布	R＝nbinrnd(R,P)	R＝nbinrnd(R,P,m)	R＝nbinrnd(R,P,m,n)
超几何分布	R＝hygernd(M,K,N)	R ＝ hygernd (M, K, N, mm)	R ＝ hygernd (M, K, N, mm,nn)
离散均匀分布	R＝unidrnd(N)	R＝unidrnd(N,mm)	R＝unidrnd(N,mm,nn)

表 4-6 常用连续型分布

分布类型	函数调用格式		
连续均匀分布	R＝unifrnd(A,B)	R＝unifrnd(A,B,mm)	R＝unifrnd(A,B,mm,nn)
正态分布	R＝normrnd(MU, SIGMA)	R＝normrnd(MU, SIGMA,m)	R＝normrnd(MU, SIGMA,m,nn)
瑞利分布	R＝raylrnd(B)	R＝raylrnd(B,m)	R＝raylrnd(B,m,nn)
指数分布	R＝exprnd(MU)	R＝exprnd(MU,m)	R＝exprnd(MU,m,nn)
β 分布	R＝betarnd(A,B)	R＝betarnd(A,B,m)	R＝betarnd(A,B,m,nn)
γ 分布	R＝gamrnd(A,B)	R＝gamrnd(A,B,m)	R＝gamrnd(A,B,m,nn)
对数正态分布	R＝lognrnd(MU, SIGMA)	R＝lognrnd(MU, SIGMA,m)	R＝lognrnd(MU, SIGMA,m,n)

续表

分布类型	函数调用格式		
χ^2 分布	R=chi2rnd(V)	R=chi2rnd(V,m)	R=chi2rnd(V,m,n)
F 分布	R=frnd(V1,V2)	R=frnd(V1,V2,m)	R=frnd(V1,V2,m,n)
t 分布	R=trnd(V)	R=trnd(V,m)	R=trnd(V,m,n)
威布尔分布	R=weibrnd(A,B)	R=weibrnd(A,B,m)	R=weibrnd(A,B,m,n)

【例 4.1.3-1】 生成正态分布的随机数向量和矩阵。

```
MU1=1;SIGMA1=2; R1=normrnd(MU1,SIGMA1)
R=normrnd(MU1,SIGMA1,1,3) %后两位的 1,3 分别是行维和列维
MU2=[1 2];SIGMA2=[3 4]; R2=normrnd(MU2,SIGMA2)
MU3=[1 2 3;4 5 6 ];SIGMA3=[7 8 9; 10 11 12]; R3=normrnd(MU3,SIGMA3)
R4=normrnd(MU3,SIGMA3,2,3)
R5=normrnd(MU3,SIGMA3,[ 2 3])
R1 =
    0.1349
R =
    -0.4086    -1.0363    0.6358
R2 =
    -3.9968     2.5013
R3 =
    1.4150    -4.6588    -9.0256
    3.0435     8.2385    14.5719
R4 =
     12.3649     8.8640    -11.3436
    -2.9178     18.7940    -11.2916
R5 =
    4.9980     7.5200     9.4072
    0.0011    13.9718    21.4830
```

4.1.4 分布的均值和方差

对于离散型随机变量 X 的分布为 $P\{X=x_k\}=p_k, k=1,2,\cdots$，有 X 的均差：

$$E(x)=\sum_{k=1}^{\infty}x_kP_k$$

若随机变量 X 是连续性的，其概率密度为 $f(x)$，则亦有 X 的均差：

$$E(x)=\int_{-\infty}^{+\infty}xf(x)\mathrm{d}x$$

关于概率分布的均值和方差函数见表 4-7 和表 4-8，他们都是以“stat”结尾。

表 4-7　**常用离散型分布**

分布类型	均值数学表达式	方差数学表达式	命令调用格式
二项分布	np	$npq, q=1-p$	[M,V]=binostat(N,P)
泊松分布	λ	λ	[M,V]=poissstat(LAMBDA)
几何分布	q/p	$q/p^2, q=1-p$	[M,V]=geostat(P)
负二项分布	$\frac{rq}{p}$	$\frac{rq}{p^2}, q=1-p$	[M,V]=nbinstat(R,P)
超几何分布	$N\frac{K}{M}$	$N\frac{K}{M}\frac{M-K}{M}\frac{M-N}{M-1}$	[M,V]=hygestat(M,K,N)
离散均匀分布	$\frac{N+1}{2}$	$\frac{N^2-1}{12}$	[M,V]=unidstat(N)

表 4-8　**常用连续型分布**

分布类型	均值数学表达式	方差数学表达式	命令调用格式
连续均匀分布	$\frac{a+b}{2}$	$\frac{b-a}{12}$	[M,V]=unifstat(A,B)
正态分布	μ	σ^2	[M,V]=normstat(MU, SIGMA)
瑞利分布	$b\left(\frac{\pi}{2}\right)^{\frac{1}{2}}$	$\frac{2-\pi}{2}b^2$	[M,V]=raylstat(B)
指数分布	μ	μ^2	[M,V]=exppstat(P, MU)
β 分布	$\frac{a}{a+b}$	$\frac{ab}{(a+b+1)(a+b)^2}$	[M,V]=betastat(A,B)
γ 分布	ab	ab^2	[M,V]=gamstat(A, B)

续表

分布类型	均值数学表达式	方差数学表达式	命令调用格式
对数正态分布	$e\left(\mu+\frac{\sigma^2}{2}\right)$	$e^{(2\mu+2\sigma^2)}-e^{(2\mu+\sigma^2)}$	[M, V] = lognstat (MU, SIGMA)
χ^2 分布	n	$2n$	[M,V]=chi2stat(NU)
F 分布	$\frac{v_2}{v_2-2}, v_2>2$	$\frac{2v_2(v_1+v_2-2)}{v_1(v_2-2)^2(v_2-4)}$, $v_2>4$	[M,V]=fstat(V1,V2) 当 $V2<3$ 时，均值无定义；当 $V2<5$ 时，方差无定义
t 分布	$v>1$ 时，均值为0；$v=1$ 时，均值不存在	$v>2$ 时，$\frac{v}{v-2}$	[M,V]=tstat(NU)
威布尔分布	$a^{\frac{1}{b}}\Gamma(1-b^{-1})$	$a^{\frac{2}{b}}[\Gamma(1+2b^{-1})\quad\Gamma^2(1+b^{-1})]$	[M,V]=weibstat(A,B)

【例 4.1.4-1】 甲乙两人打靶，所得分数分别为 X_1，X_2，它们的分布律是：

X_1	0	1	2
P_k	0	0.2	0.8

X_2	0	1	2
P_k	0.6	0.3	0.1

求他们的均值和方差。

```
    x=[0 1 2];p1=[0 0.2 0.8];
    x=[0 1 2];p2=[0.6 0.3 0.1];
    [m1,v1]=binostat(x,p1);M1=sum(m1), V1=sum(v1)
    [m2,v2]=binostat(x,p2); M2=sum(m2), V2=sum(v2)
M1 =          %均值
    1.8000
V1 =          %方差
    0.4800
M2 =          %均值
    0.5000
V2 =          %方差
    0.3900
```

【例 4.1.4-2】 求威布尔分布的均值和方差。

```
x=(0.5:0.1:5);y=(1:0.04:2);[X,Y]=meshgrid(x,y);%形成xoy平面网线栅格
Z=weibstat(X,Y);                %形成威布尔分布曲面坐标
```

```
[c,h]=contour(x,y,Z,[0.4 0.6 1.0 1.8]); %形成等值线
clabel (c) ;                   %对等值线进行标识,结果如图 4.1.4-1 所示
```

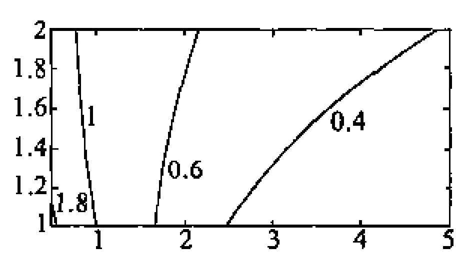


图 4.1.4-1　对等值线标识

4.2 参数估计

参数估计是统计推断问题,即当总体分布的数学形式已知,用有限个参数表示的估计的问题。它可分为点估计和区间估计两个方面。在参数模型中,最常用的还是一种参数估计方法——极大似然法。MATLAB 的统计工具箱用极大似然法给出了常用的概率分布的参数的点估计和区域估计值。

表 4-9　参数估计函数

分布类型	命令调用格式	命令调用说明
最大似然估计(通用函数)	phat=mle('dist',data) [phat,pci]=mle('dist',data) [phat,pci]=mle('dist',data,alpha) [phat,pci]=mle('dist',data,alpha,pl)	'dist':函数名 data:数据样本 alpha:置信度 phat:参数估计结果 pci:置信区间计算结果 表中其他各函数的 x,alpha,phat 和 pci 的含义与此相同。
二项分布	phat=binofit(x,n) [phat,pci]=binofit(x,n) [phat,pci]=binofit(x,n,alpha)	n:总试验次数 phat:试验成功的概率参数 p 的估计值

续表

分布类型	命令调用格式	命令调用说明
泊松分布	lambdahat=poissfit(x) [lambdabat,lambdaci]=poissfit(x) [lambdabat, lambdaci] = poissfit (x, alpha)	lambdahat:参数λ的估计值 lambdaci:参数λ估计的置信区间
均匀分布	[ahat,bhat]=unifit(x) [ahat,bhat,aci,bci]=unifit(x) [ahat,bhat,aci,bci]=unifit(x,alpha)	ahat:参数a的估计值 aci:参数a估计的置信区间 bhat:参数b的估计值 bci:参数b估计的置信区间
正态分布	[muhat, sigmahat, muci, sigmaci] = normfit(x) [muhat, sigmahat, muci, sigmaci] = normfit(x,alpha)	muhat:参数μ的估计值 muci:参数μ估计的置信区间 sigmahat:参数σ的估计值 sigmaci:参数σ估计的置信区间
指数分布	muhat=expfit(x) [muhat,muci]=expfit(x) [muhat,muci]=expfit(x,alpha)	muhat:参数μ的估计值 muci:参数μ估计的置信区间
β分布	phat=betafit(x) [phat,pci]=betafit(x,alpha)	x:数据样本 phat:参数a,b的估计结果矢量[a,b]。置信区间结果pci以2*2矩阵的形式给出,其第一列为参数a的置信下界和上界,第二列为参数b的置信下界和上界,以下类同
γ分布	phat=gamfit(x) [phat,pci]=gamfit(x) [phat,pci]=gamfit(x,alpha)	phat:参数a,b的估计值[a,b] pci:参数a,b估计的置信区间
威布尔分布	phat=weibfit(x) [phat,pci]=weibfit(x) [phat,pci]=weibfit(x,alpha)	phat:参数a,b的估计值[a,b] pci:参数a,b估计的置信区间

【例4.2-1】 算出泊松分布的极大似然估计和置信区间。

```
r=poissrnd(5,10,3);
[L,Lci]=poissfit(r)
L =
     5.9000     5.8000     5.7000
Lci =
     4.4000     4.4000     4.3000
     7.5000     7.3000     7.2000
```

4.3 描述性统计

无论在工程界还是科技界，人们会遇到成千上万的数据，这样就总是想用少量的包含相关信息的数值去描述数据样本总体的规律。描述性统计就是搜集、整理、加工和分析统计数据，使数据系统化、条理化，以显示出数据资料的趋势、特征和数量关系。它是统计推断的基础，因而在统计工作中经常使用。MATLAB 提供了 30 多个有关函数，列在表 4-10 中。

表 4-10　描述性统计函数

函　数	功　能	函　数	功　能
max	最大元素	mad	平均绝对偏差
min	最小元素	range	样本极差
prod	元素之积	std	标准差
sortrows	按行升序排列	moment	任意阶中心距
sum	元素之和	cov	协方差矩阵
cumtrapz	元素累计数值积分	nanmax	求忽略缺失数据的最大值
cumsum	元素累计和	nanmean	求忽略缺失数据的平均值
cumprod	元素累计乘积	nanmedian	求忽略缺失数据的中位数
sort	使元素递增排序	nanmin	求忽略缺失数据的最小值
geomean	几何均值	nanstd	求忽略缺失数据的标准差
harmmean	调和均值	nansum	求忽略缺失数据的和
mean	算术平均值	corrcoef	计算相关系数
median	中位元素	prctile	样本的经验分位数
trimmean	修正的样本均值	kurtosis	计算样本峰度
var	方差	skewness	计算样本偏度
iqr	内四分位数间距	bootstrp	通过对数据重新采样进行自助统计

【例 4.3-1】 用不同函数考察数据样本 x 的中心位置度量，及对野值 1 000 的敏感程度。

```
x=[1000 ones(1,7)];                %中心位置度量
```

```
LO=[ geomean(x) harmmean(x) mean(x) median(x) trimmean(x,25)]
%散步度量
ST=[iqr(x) mad(x) range(x) std(x) var(x)]
LO =
   2.3714    1.1427    125.8750    1.0000    1.0000
ST =
   0    218.5313    999.0000    353.1998
```

说明：均值是对位置简单的估计量。若数据样本符合正态分布，均值应是很好的，但实际上由于操作原因，输入时会带进一些野值，这样会使样本均值偏离正常值很远。可见，前两个函数对野值的敏感性较强。

trimmean(x,25)是计算剔除数据观测量中最高25%和最低25%的数据后的均值。

散步度量是样本中数据偏离数据中心的程度。iqr 对野值的抗干扰能力很强。

【例 4.3-2】 对5列随机 ***A*** 阵进行统计。

```
randn('seed',0);
A=randn(50,5);            %正态分布
SUM=sum(A)                %按列求和
AMAX=max(max(A))          %先按列后按行求最大值
AMED=median(A)            %按列求中位元素
AMEAN=mean(A)             %按列求平均值
ASTD=std(A)               %找A各列的标准差
SUM =
   2.6087    4.6660    -0.2924    2.9283    -8.8239
AMAX =
   2.5659
AMED =
   0.1115    0.2338    0.1242    -0.0402    -0.1712
AMEAN =
   0.0522    0.0933    -0.0058    0.0586    -0.1765
ASTD =
   0.9573    0.9592    1.1753    0.9480    0.9977
```

【例 4.3-3】 在随机阵 ***X***,***Y*** 上求协方差和相关系数。

```
rand('seed',0)
X=rand(10,1);
Y=rand(10,1);
CX=cov(X); CY=cov(Y);
```

```
        Cxy=cov(X,Y)
        PX=corrcoef(X)
        Pxy=corrcoef(X,Y)
CX =
   0.0676
Cxy =
     0.0676   -0.0044
    -0.0044    0.0823
PX =
     1
Pxy =
     1.0000   -0.0589
    -0.0589    1.0000
```

【例 4.3-4】 X,Y 中的每个列都作为独立变量的记录看待时，求 X,Y 的互协方差阵。

```
        xbar=mean(X);[mx,nx]=size(X);   %X 的阶数
        ybar=mean(Y);[my,ny]=size(Y);
        CXY=(X-ones(mx,1)*xbar)'*...
(Y-ones(my,1)*ybar)/(mx-1)
        xv=sqrt(diag(CX));
        yv=sqrt(diag(CY));
        PXY=CXY./(xv*yv')
CXY =
    -0.0764
PXY =
    -0.7418
```

4.4 方差及回归分析

4.4.1 概　　述

方差分析、回归分析是分析试验(或观测)数据的一种方法，是数理统计中的一个重要分支。它可通过数据的分析，弄清与研究对象有关的各个因素以及各因素之间的交互作用对该对象的影响。通常可以表示为以下形式：

$$y = x\beta + \varepsilon \tag{4.4.1-1}$$

其中：y——$n\times1$ 观测值向量；

x——模型的系数矩阵；

β——$P\times1$参数向量；

ε——$n\times1$随机干扰矢量。

这部分内容较多，MATLAB统计工具箱提供了最常用的一些功能。在方差分析方面，MATLAB给出了单因素方差分析、双因素方差分析函数；在回归分析方面，给出了多元线性回归、多项式回归和逐步回归等函数；工具箱还提供了图形化的二次响应曲面模型。工具箱所提供的线性模型分析的主要函数见表4-11。

表4-11 线性模型函数表

函数分类	函数	功能
方差分析	anova1	单因素方差分析
	anova2	双因素方差分析
回归分析	regress	多重线回归
	lscov	给定方差矩阵回归
	ridge	岭回归
	stepwise	逐步回归GUI
多项式回归	ployfit	多项式拟合
	ployval	多项式预测
	ployconf	给出置信区间的多项式预测
二项响应曲面模型	rstool	二次响应曲面模型的交互式图形工具

4.4.2 方差分析

在科学研究和工业过程中，影响试验或生产的速度和质量的因素是很多的，也很复杂。为了解决这一问题，就应当进行分析，找出有显著影响的那些因素，即在各因素作用下分析相应的正态性和等方差性。

1. 单因素方差分析

单因素方差分析的基本问题，是比较和估计多个等方差正态总体的均值。其模型为：

$$y_{ij}=\alpha_{oj}+\varepsilon_{ij} \tag{4.4.2-1}$$

其中：y_{ij}——观测值矩阵；

α_{oj}——各列为组均值的矩阵（α_{oj}表示α中第j列的所有行）；

ε_{ij}—— 随机扰动矩阵。

单因素方差分析表有四列：SS 平方和、自由度 df、均方值 MS、统计量 F。

主要函数格式：

P＝anoa1(X)

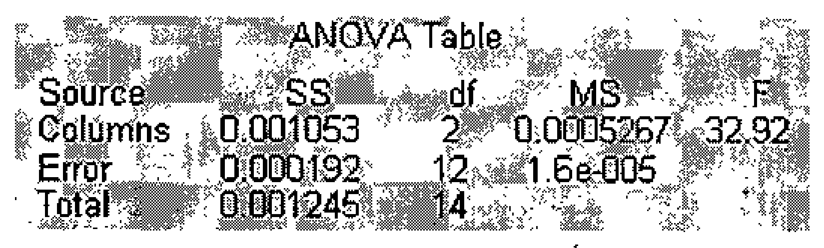
X 为两列或多列数据样本。

【例 4.4.2-1】 设有三台同样规格的机器，用来生产厚度为 1/4 厘米的铝板。现在要了解机器产品的平均厚度是否相同，取样测至千分之一厘米，结果为表4-12所示。

表 4-12　　测试三台机器生产的铝板

Ⅰ	Ⅱ	Ⅲ
0.236	0.257	0.258
0.238	0.253	0.264
0.248	0.255	0.259
0.245	0.254	0.267
0.243	0.261	0.262

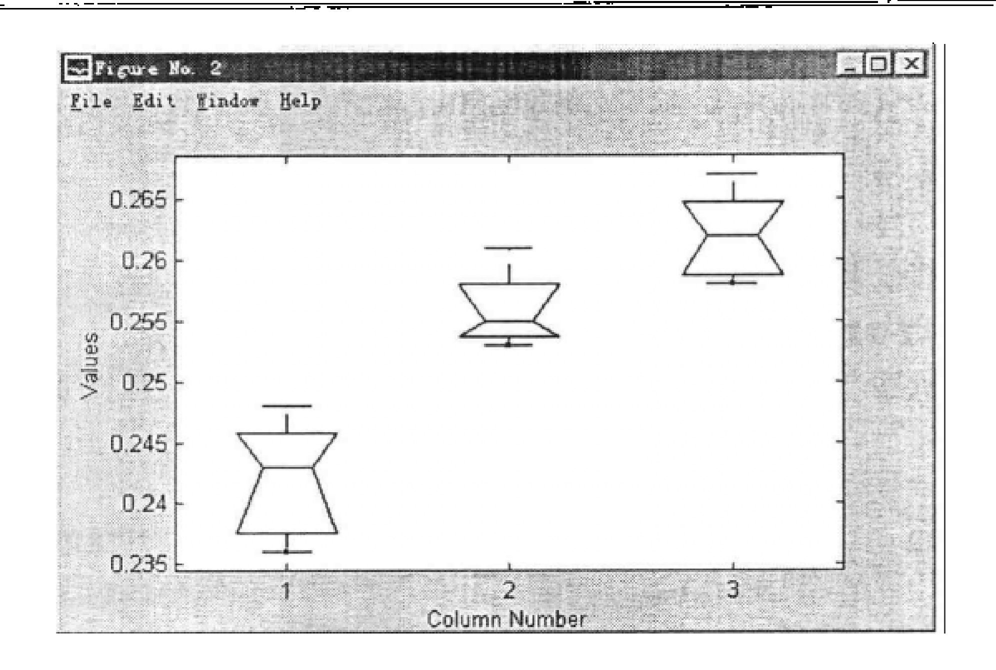
```
X=[0.236   0.257   0.258;
   0.238   0.253   0.264;
   0.248   0.255   0.259;
   0.245   0.254   0.267;
   0.243   0.261   0.262]
P=anova1(X)
P =
   1.3431e-005
```


ANOVA Table

Source	SS	df	MS	F
Columns	0.001053	2	0.0005267	32.92
Error	0.000192	12	1.6e-005	
Total	0.001245	14		

ANOVA 表具有四列：平方和 SS、自由度 df、均方值 SS/df、F 统计量。可使用 F 统计量来进行假设检验，判断样本的各列数据是否为同一均值。

由计算结果可知，观察所提供的 X 的随机数据样本，给出各列均值相同结论



的概率小于十万分之二，差异显著。机器I与另外两台机器生产的铝板都要薄一些。

2. 双因素方差分析

双因素方差分析是一种量因素、多水平析因试验数据的统计分析方法，目的在于确认来自不同组的数据是否具有相同的均值。

数学模型：

$$y_{ikj} = \mu + \alpha_{oj} + \beta_{io} + \gamma_{ij} + \varepsilon_{ijk} \qquad (4.4.2\text{-}2)$$

其中：y_{ijk}——观测值矩阵；

μ——样本总均值；

α_{oj}——各列元素为组均值的矩阵；

β_{io}——各行元素为组均值的矩阵；

γ_{ij}——互相作用项(矩阵)；

ε_{ijk}——随机干扰矩阵。

主要函数格式：

P＝anova2(X)

X 为两列或多列数据样本。

【例 4.4.2-2】　一火箭使用了六种燃料、三种推进器作射程试验。每种燃料与每种推进器的组合作一次试验，得火箭射程(单位：海里)如表 4-13 和表 4-14 所示：

表 4-13　某火箭的射程试验

推进器(B)		B1	B2	B3
燃料(A)	A_1	33.3	34.5	37.4
	A_2	33.4	34.8	36.8
	A_3	32.9	33.8	37.6
	A_4	32.6	33.4	36.6
	A_5	32.5	33.7	37.0
	A_6	32.0	33.9'	36.7

```
x=[33.3    34.5    37.4;
           33.4    34.8    36.8;
           32.9    33.8    37.6;
           32.6    33.4    36.6;
           32.5    33.7    37.0;
   32.0    33.9    36.7]
   P=anova2(x,3)
P =
   0.0000    0.0013    0.7093
```

表 4-14　某火箭的射程试验的 ANOVA 表

Source 方差源	SS 平方和	Df 自由度	MS 均方值	F 均方比
Columns	56.88	1	28.44	238.1
Rows	2.067	2	2.067	17.31
Interaction	0.08444	2	0.04222	0.3535
Error	1.433	12	0.1194	
Total	60.47	17		

第一行：由各列均值而产生的变化。

第二行：由各行均值而产生的变化。

第三行：行和列的交互作用而导致的变化。

用 F 统计来假设检验海里数与推进器种类、燃料以及推进器种类与燃料之间作用的依赖关系。anova2 则直接给出检验的 p 值。结果表明，推进器的 p 值为 0.000，明显表示海里数随推进器种类的变化而变化；燃料的 p 值为 0.0013，说明海里数也因工厂的不同而存在差异。而推进器与燃料交互作用项为 0.7093，表明

交互作用项几乎不起作用。

注意：anova2 所返回的 p 值依赖于对模型方程中随机干涉量所作的假设。要获得正确的 p 值，要求干扰是独立正态分布，且方差为常数。

4.4.3 回归分析

工程和科学计算中，变量之间往往存在着某种关系。回归分析就是用统计数据来寻求变量间关系的近似表达式，即经验公式，并利用所得公式进行统计描述、分析和推断，解决预测、控制和优化问题。

多重线性回归

假设回归函数的未知参数和回归变量为线性的多元回归，其变量关系为：

$$y=X\beta+\varepsilon$$

$$\varepsilon\sim N(0,\sigma^2I)$$

其中：y—$n*I$ 观测值矢量；

X—$n*p$ 回归矩阵；

β—$p*I$ 参数矢量；

ε—$n*I$ 随机干扰矢量。

多重线性回归主要内容是：

1）从对 X、Y 的观测数据出发，求出回归系数 β 的估计量 $\hat{\beta}$。

2）检验回归方程及各回归系数的显著性。

3）利用经验回归方程 $\hat{y}=X\hat{\beta}+\varepsilon$ 进行预测和控制。

MATLAB 函数格式：

```
b=regress(y,X)
[b,bint]=regress(y,X,alpha)
```

函数式 b 可给出观测 y 和回归矩阵 X 的最小二乘拟合 β 的结果。

b：β 的估计；

矢量 bint：β 的 95%置信区间(p * 2 矢量 rint)；

r：残差；

矢量 rint：每个残差的 95%置信区间(n * 2 矢量 rint)；

矢量 staus：可给出回归的 R^2 统计量和 F 以及 p 值。

【例 4.4.3-1】 设有实验数据回归矩阵 *x* 及对应观测值 *y* 数据如下，求最小二乘拟合表达式、估计参数矢量及置信区间矢量。

```
x=[1 1 1 1 1 1 1 1 1 1;
   1 2 3 4 5 6 7 8 9 10]';
y=[11.73 12.27 13.45 14.62 15.93 16.14 17.43 18.68 19.53 20.31]';
```

```
    [b,bint]=regress(y,x,0.05)
b =
    10.5813
     0.9868
bint =
    10.1889    10.9737
     0.9236     1.0501
```

4.5 非线性回归

非线性回归即响应曲面方法是采用经验公式模型去逼近输出和输入之间的关系。在工程计算和科学应用中，常常用这种经验模型去表达一些事物的发展规律或描述相关理论。但这些模型参数是未知的，往往很难拟合，需要首先估算未知参数的初始值，然后进行迭代，而每次迭代都会修正当前的估计值直至算法收敛为止。

假设拟合的非线性模型：

$$y = F(x,\beta) + \epsilon \tag{4.5-1}$$

y—$n*I$ 观测矢量 y；

F—x,β 的函数；

x—$n*p$ 输入变量矩阵；

β—$p*I$ 预测的未知参数矢量；

ϵ—$n*I$ 随机干扰矢量。

实现上述拟合的 MATLAB 函数：

函　数	说　明
nlinfit	非线性最小二乘(牛顿法)
nlintool	对于非线性模型预测交互图形
nlpredci	预测置信区间
nlparci	参数置信区间
nnls	非负最小二乘拟合

【例 4.5-1】 反应力学中的 Hongen-Watson 非线性模型

$$rate = \frac{\beta_1 \cdot x_2 - x_3/\beta_5}{1 + \beta_2 \cdot x_1 + \beta_3 \cdot x_2 + \beta_4 \cdot x_3} \tag{4.5-2}$$

1. Hongen-Watson 模型拟合

nlinfit 函数是用于获取非线性模型的参数估算值，并返回最小二乘参数估算，即求观察反应与其拟合值差的最小二乘，采用了全局收敛的 Levenberg-Marquardt 修正的 Gauss-Newton 算法。

nlinfit 需要输入数据和未知参数的初始估计值，并返回预测反应。

(1) 用编辑器编程 hougen. m，定义式(4. 5-2)。

```
function yhat = hougen(beta,x)
% YHAT = HOUGEN(BETA,X) 提供反应速度的预测值
% YHAT 是未知参数矢量 BETA 的函数
% X 为 3 列的数据矩阵
b1 = beta(1);
b2 = beta(2);
b3 = beta(3);
b4 = beta(4);
b5 = beta(5);
x1 = x(:,1);
x2 = x(:,2);
x3 = x(:,3);
yhat = (b1 * x2 - x3/b5)./(1+b2 * x1+b3 * x2+b4 * x3);
```

(2) 在 MATALB 命令窗口利用 nlifit. m 函数对反应力学数据进行拟合。

```
load reaction   %装入数据文件其中含有变量 beta,rate,xn,
                %model,reactants,yn
betahat = nlinfit(reactants,rate,'hougen',beta)
betahat =
1.2526
0.0628
0.0400
0.1124
1.1914
```

说明：beta——5×1 参数的初始观测矢量；

rate——13×1 反应率观测矢量；

xn——反应物名的字符矩阵；

'model'——非线性函数名；

reactants——13×1 反应物矩阵；

yn——反应结果名的字符串。

这里 nlifit 有两个可选的输出，即结果的残差和雅可比矩阵。残差为拟合值与

观测值之差；雅可比矩阵则类似于标准线性回归模型中矩阵X。这两个可选的输出可用于参数估计值和反应结果预测值的置信区间计算。

2. 参数估计的置信区间

使用函数nlparci可获得上述参数估计值betahat的95%置信区间。

```
[betahat,f,J]= nlinfit(reactants,rate,'hougen',beta);
betaci=nlparci(betahat,f,J)
betaci =
-7467    3.2519
-0377    0.1632
-0312    0.1113
-0609    0.2857
-7381    3.1208
```

3. 结果预测值的置信区间

使用函数nlpredci可获得问题1中反应结果预测值的95%置信区间。

```
[yhat,delta]=lpredci('hougen',reactants,betahat,f,J)
opd=[rate yhat delta]
opd =
8.5500     8.2937    0.9178
3.7900     3.8584    0.7244
4.8200     4.7950    0.8267
0.0200    -0.0725    0.4775
2.7500     2.5687    0.4987
14.3900    14.2227   0.9666
2.5400     2.4393    0.9247
4.3500     3.9360    0.7327
13.0000    12.9440   0.7210
8.5000     8.2670    0.9459
0.0500    -0.1437    0.9537
11.3200    11.3484   0.9228
3.1300     3.3145    0.8418
```

结果矩阵opd中的第一列为观测比率，第二列为预测值，其95%置信区间为第二列数据±第三列数据。注意，置信区间含有每种情况的观测值。

4.6 假设检验

假设检验是统计推断的基本问题之一，主要是根据样本的信息来判断总体分

布是否具有指定的特征，确定关于样本总体特征的判断是否合理的过程。按一定规则（检验准则）根据样本所作假设 H 是否成立，以决定是接受还是否定 H。假设检验的判断和结论是根据样本做出的，则具有“概率性”。

例如，现在武汉市西红柿的平均价格为每斤1.15元，怎样决定这个说法是正确的呢？

一个简单的方法就是在市内抽取少数的菜场，查明其价格以及销售量，将统计结果与1.15元的价格相比较。如果算出来的平均价格与假设有差异，那么，这是否就说明原来的假设就不正确呢？假设检验就是解决此类问题的方法。

相关概念：

● 零假设——即初始判断。此例中，零假设是平均价格每斤1.15元，可表达为：

$$H_0:\mu = 1.15 \tag{4.6-1}$$

备选假设（或对立假设）：

$$H_1:\mu > 1.15;H_1:\mu < 1.15;H_1:\mu \neq 1.15. \tag{4.6-2}$$

● 显著性水平——是与支持对立假设而拒绝零假设的确定度有关的量。在小样本的前提下，不能肯定自己的结论，所以事先约定，如果观测到符合零假设的样本值的概率小于显著性水平 α，则拒绝零假设。典型的显著性水平 $\alpha=0.05$。如果要减少犯错误的可能，可取更小的值。

● p-值——在假定零假设为真的条件下，观测给定样本结果的概率值。如果 p-值小于 α，则拒绝零假设，反之则并非如此。如果 p-值大于 α，则并不能肯定就接受零假设。此时，您只是没有足够的证据来拒绝假设。

假设检验的输出包括置信区间。不严格地说，置信区间可能是包含真实的假设量的可选概率的值的范围。例如1.15在均值为 μ 的95%置信区间内，即意味着在0.05的显著性水平下，不能拒绝零假设。

MATLAB相关函数列在表4-15中，因篇幅的关系仅介绍关于主要函数ZTEST的用法。

Z试验是用方差的单样本均值与实际观察比较的假设试验。函数格式为：

```
h=ztest(x,m,sigma)
h=ztest(x,m,sigma,alpha)
[h,sig]=ztest(x,m,sigma,alpha,tail)
```

Z试验是在某显著水平下检验正态分布的样本(x)是否具有均值m和标准差sigma，缺省值显著水平alpha为0.05，tail是0。

sig是与统计量有关的，在假设x均值等于 μ 或m时，观测预定结果的概率。

H=0 则“不能拒绝在alpha显著水平下的零假设”；

H=1 则“拒绝在 alpha 显著水平下的零假设”。
零假设:意味着“x 等于均值 M”。
对于 TAIL = 0，意味着“x 不等于均值 M”;
对于 TAIL = 1，意味着“x 大于均值 M”;
对于 TAIL = −1，意味着“x 小于均值 M”。

【例 4.6-1】 有西红柿价格数据向量,共 40 个数据,分别为 2001 年 1～3 月的价格。假设武汉各菜场西红柿价格的标准差为每斤 0.14 元。用 T 检验来判断量假设:1 月份每斤西红柿的平均价格 1.15 元。

```
price=[1.19 1.18 1.17 1.15 1.15 1.15 1.16 1.22 1.12 1.18
       1.21 1.21 1.15 1.20 1.22 1.22 1.16 1.20 1.18 1.13
       1.09 1.20 1.12 1.23 1.19 1.21 1.12 1.09 1.17 1.17
       1.13 1.17 1.14 1.20 1.09 1.16 1.09 1.18 1.18 1.25]
[h,pvalue,ci]=ztest(price,1.15,0.14)
h =
    0
pvalue =
    0.4097
ci =
    1.1249    1.2116
```

结果说明:h=0 则不能拒绝零假设。1.15 元假设合理,95%置信区间包含了 1.15 元人民币。如果将标准差改为 0.02,则 h=1,会有不同的结论。

表 4-15　假设检验表

函　数	功　能
ranksum	威尔科克秩和检验
signrank	威尔科克符号秩检验
signtest	成对样本的符号检验
ttest	单样本 t 检验
ttest2	双样本 t 检验
ztest	z 检验

4.7 统计绘图

统计工具箱在 Matlab 绘图功能上又增添了一些统计图形表现函数,以直观地

表现样本及其统计量的内在联系和规律。主要的函数见表4-16。

表4-16　　统计绘图表

函　　数	功　　能
boxplot	box 图
errorbar	误差条图
fsurfht	函数的交互轮廓图
gline	交互直线绘制
gname	交互点标记
lsline	绘制数据图
normplot	正态概率图
pareto	帕累托图
qqplot	分位数-分位数图
rcoplot	回归残差图
refcurve	参考多项式
refline	参考线
surfht	交互内插轮廓图
weibplot	威布尔图用以确定数据是否为威布尔分布的图形

1. box 图

格式：boxplot(X,notch, 'sym', vert, whis)

说明：除X以外均可缺省。用于描述数据样本分布，也可通过图形来比较多个样本的均值。

- boxplot(X)为X中的每列数据绘制一个box图。
- boxplot(X,notch)当notch取1可绘制带切口的box图；缺省时(即取notch为0),box图无切口。
- boxplot(X, notch,'sym',vert)中的vert控制box图水平放置还是垂直放置。当vert=0时，box图水平放置，当vert=1(缺省)时，box图垂直放置。
- boxplot(X,notch,'sym',vert,whis)中的whis定义虚线的长度为内四分位间距(IQR)的变量(缺省情况为1.5 * IQR)。如果whis=0,则box图用'sym'规定的标记显示"盒子"外所有的数据。

【例 4.7-1】 画出 box 图，以显示两个数据样本的分布。

```
x=normrnd(150,1,100,1);y= normrnd(160,1,100,1) ;f=[x y];
boxplot(f,1)
```

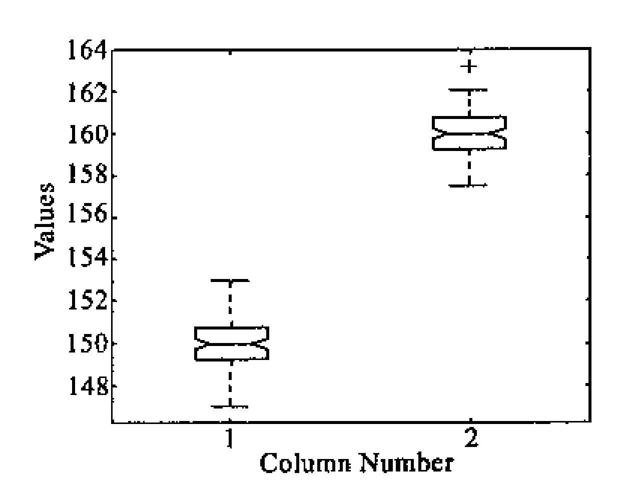


图 4.7-1 x,y 向量的数据分布 box 图

如图 4.7-1 所示，box 图为一“盒子”图形，其基本特征为：

● “盒子”的上底和下底间为内四分位间距；x 向量“盒子”上下的两条线分别为样本的 50%和 50%分位数；y 向量“盒子”上下的两条线分别为样本的 40%和 60%分位数。

● “盒子”中间的线为样本中位数。如果中位数不在“盒子”中间，则表明样本存在一定的偏度。

● 虚线贯穿“盒子”上下，显示了样本的其余部分(除非有野值)。假设没有野值，样本的最大值为虚线的顶端；样本的最小值为虚线的底端。缺省时，野值为距离盒子顶端和底端超过 1.5 倍内四分位间距的点。在图中，野值为超出虚线底端的点。

● “+”表示数据的一个野值。

● “切口”是样本中位数的置信区间。缺省时，box 图是没有切口的。

2. 正态概率图

格式：normalplot(X)

正态概率图是检查数据 X 是否为正态分布。因为，许多统计过程首先假设数据分布是正态的，所以正态概率图可提供证明的依据(参见图 4.7-2)。

【例 4.7-2】 检查数据 X 是否为正态分布。

```
x = normrnd(10,1,25,1);
normplot(x)
```

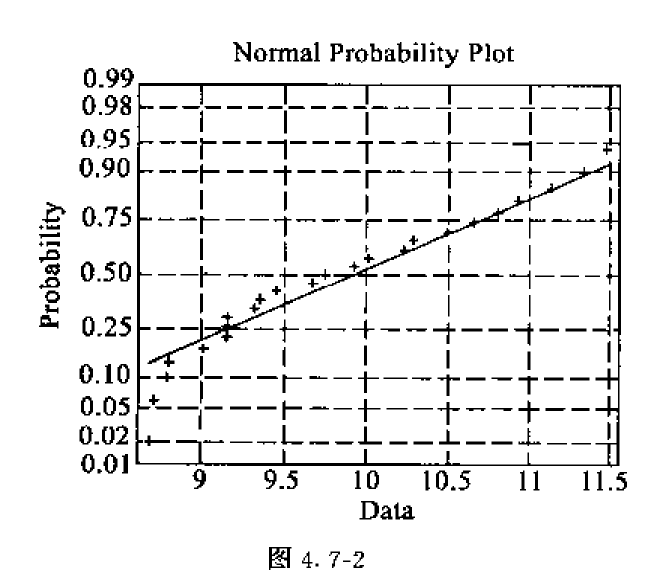


图 4.7-2

说明：对于矩阵X，normalplot(X)为其每列数据显示一条线。图形以符号“+”标识样本数据。叠加在数据点上的实线是X的每列数据的第一和第三分位间(25%至75%的数据点)的连线，代表样本的鲁棒线性拟合。虚线延伸至样本的两端，有助于评价数据的线性程度。方法是用谷底到第一分位的垂直距离去与第一分位到第二分位的距离比较；同样，再用谷顶到第三分位的距离去与第三分位到第二分位的距离比较。如果数据点接近直线，则正态分布的假设是合理的而其他概率分布的数据会表现出不同的弯曲(参见图4.7-3)。

```
x = exprnd(10,100,1);
normplot(x)
```

3. 分位数-分位数图

格式：qqplot(X,Y)

分位数-分位数图可用来检查两个样本是否服从同一分布。

【例4.7-3】 检查两个样本是否服从同一分布。

```
x = poissrnd(10,50,1);
y = poissrnd(5,100,1);
qqplot(x,y);
```

说明：同正态概率图相似。如果两个样本来源于同一分布，那么图中的数据样本就接近直线。若X和Y为矩阵，则为它们的每列数据绘制单独的曲线。图中样本数据以“+”符号表示，并将位于第一分位数(25%)和第三分位数间(75%)的数据拟合绘制成一条线，为两个样本顺序统计量的鲁棒线性拟合。虚线延伸至样本的两端，以帮助用户评估数据的线性程度(如图4.7-4)。

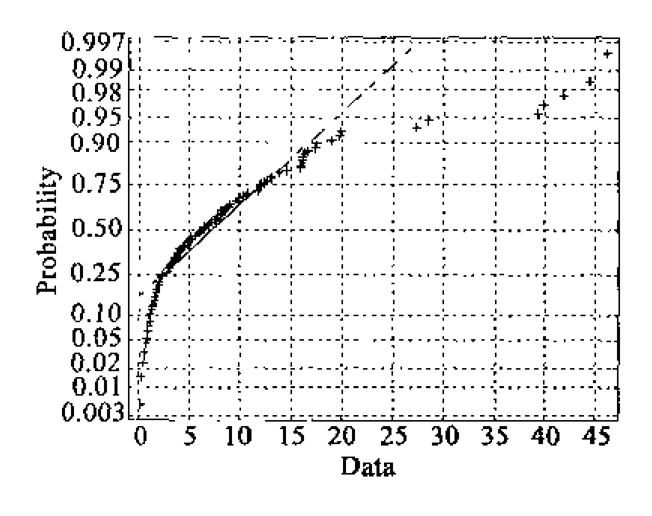


图 4.7-3

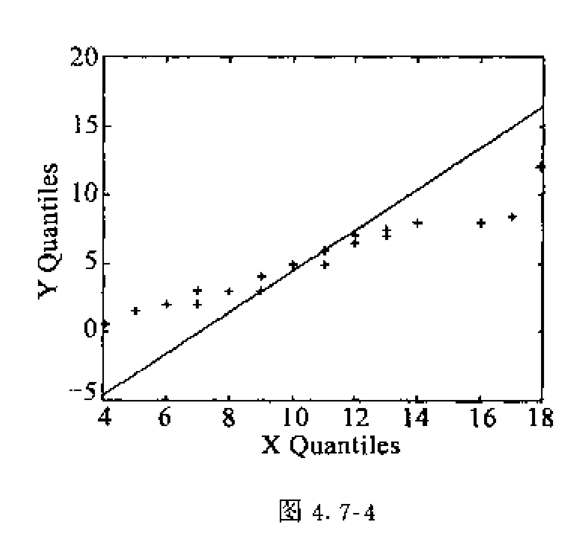


图 4.7-4

4.8 质量评估(工序管理)

工序管理是指运用数理统计的一系列理论和方法对生产过程或产品质量的管理和控制。一件产品的加工往往需要若干工序完成,每个工序出了问题都会影响产品的质量,因此在生产过程的不同工序中采用各种抽样和验收方法、各种统计估计比较和分析方法,产品质量和生产过程的管理控制图以及其他各种描述统计图和分析图等都是统计质量管理常用的方法和工具,其核心是管理图。这种方法操作起来很简单,易于在生产过程中实施。统计工具箱提供了主要的一些统计质量管理图

函数,见表4-17。

表4-17 统计工序管理

函　数	功　能
xbarplot	定点观测平均偏差曲线
schart	标准差时间曲线
ewmaplot	指数加权滑动平均图
capable	工序性能指数
capaplot	工序性能图
histfit	直方图和正态密度曲线
normspec	绘制规定区间的正态密度曲线

1. 定点观测平均偏差曲线(Xbar Charts)

定点观测平均偏差曲线是描述加工零件的粗糙度,即在等间距段上测出不同偏差的算术平均值。每一零件测得平均偏差可用 R 表示:

$$R = \frac{1}{n}\sum_{i=1}^{n}|y_i| \tag{4.8-1}$$

函数格式:xbarplot(DATA,conf,specs)

DATA 是存放观测数据的变量;conf(可缺省)确定置信区间的管理限,在图上是 UCL、LCL 两根直线;specs(可缺省)是确定上下曲线界限的两个元素矢量。

【例 4.8-1】 假设测量一个活塞的粗糙度偏差为 0.5μm,测量是每个活塞取 4 点,共 36 个活塞,数据存于 Parts.mex 文件的 runout 变量中。

```
load parts;
conf = 0.99;spec = [-0.5 0.5];
xbarplot(runout,conf,spec)
xlabel ('sample');
ylabel ('Measurements');title('Xbar Charts');
```

图中上下两根直线为粗糙度标准,中线为所有活塞观测数据的平均偏差。

2. 定点观测标准偏差曲线(S Charts)

定点观测标准偏差曲线是描述加工零件在等间距上测出的标准偏差,标准偏差是描述一个工序加工尺寸的变动量。

函数格式:schart(DATA,conf,specs)

变量含义同上 xbarplot。

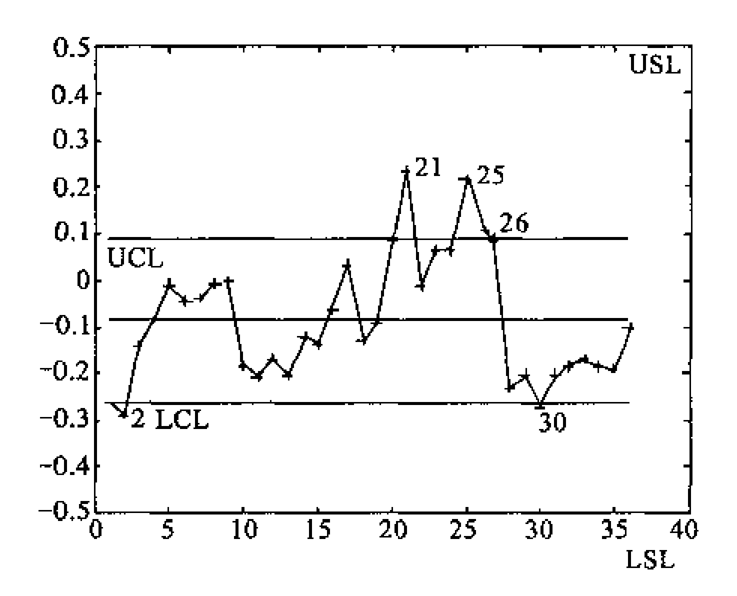


图 4.8-1 零件尺寸的定点观测平均偏差曲线

【例 4.8-2】 测量【例 4.8-1】中活塞的标准差。

```
load parts;
    schart(runout)
```

结果参见图 4.8-2,从图中可看出不同工序的系统变化。

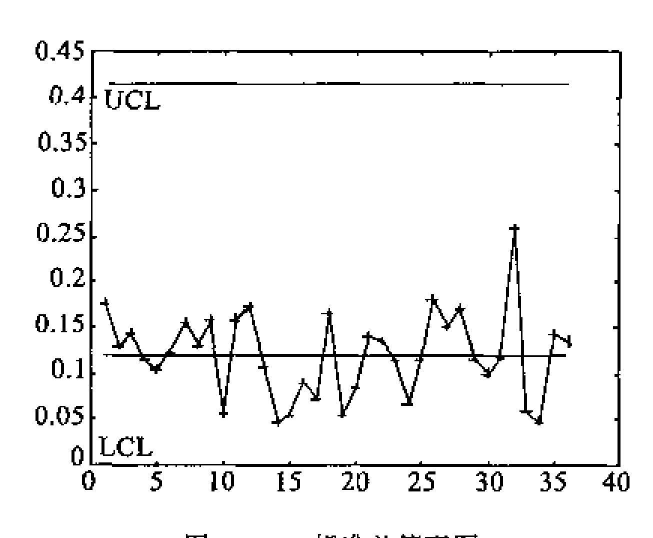


图 4.8-2 标准差管理图

习 题 四

1. 有一繁忙的汽车站,有大量汽车通过,设每辆汽车在一天的某段时间内出事故的概率为0.0001。在某天的该段时间内有1000辆汽车通过,问出事故的次数不小于2的概率是多少?(利用泊松定理来计算)

2. 一电话交换台每分钟的呼唤次数服从参数为4的泊松分布,求(1)每分钟恰有8次呼唤的概率;(2)每分钟的呼唤次数大于10的概率。

3. 某机器生产的螺栓的长度(cm)服从参数为 $\mu=10.05$, $\sigma=0.06$ 的正态分布,规定长度在范围 10.05 ± 0.12 内为合格产品,求一螺栓为不合格的概率是多少?

4. 工厂生产的电子管的寿命 x(以小时计)服从参数为 $\sigma=160$, μ 的正态分布,若要求 $p\{120<x\leqslant200\}\geqslant0.80$,允许 σ 最大为多少?

5. 设电流 I 是一个随机变量,它均匀分布在9~11安之间。若此电流通过2欧的电阻,求功率 $W=2I^2$ 的概率密度。

6. 设某种型号的电子管的寿命(以小时计)近似地服从 $N(160,20^2)$ 分布。随机地选取4只,求其中没有一只寿命小于180小时的概率。

7. 卡车装运水泥。设每袋水泥的重量 X 是一个随机变量,它服从正态分布,其数学期望为50公斤,均方差为2.5公斤,问装多少袋水泥能使总重量超过2 000公斤的概率为0.05。

8. 已知某种白炽灯泡寿命服从正态分布。在某星期所生产的该种灯泡中随机抽取10只,测得其寿命(以小时计)为:

1067 919 1196 785 1126 936 918 1156 920 948

设总体参数都为未知,试用极大似然法估计这个星期中生产的灯泡能使用1 300小时以上的概率。

9. 随机地从一批钉子中抽取16枚,测得其长度(以厘米计)为:

2.14 2.10 2.13 2.15 2.13 2.12 2.13 2.10

2.15 2.12 2.14 2.10 2.13 2.11 2.14 2.11

10. 某批矿砂的5个样品中的镍含量经测定为:

x(%) 3.25 3.27 3.24 3.26 3.24

设测定值服从正态分布,问在 $\alpha=0.01$ 下能否接受假设:这批矿砂的镍含量为3.25。

11. 下表数据是退火温度 x(℃)对黄铜延性 y 效应的试验结果,y 是以延长长度计算的,且设 y 为正态变量。

x(℃)	300	400	500	600	700	800
y(℃)	40	50	55	60	67	70

求 y 对 x 的线性回归方程。

第五章 图形显示

在工程和科学计算时，人们眼前常常会堆砌着大量的数据，因而很难直接从它们之中找到内在规律及其含义。依据数据画出图形有时加上颜色可使人们便于找到问题的本质、错误的数据，甚至领悟出抽象的理论。因此，数据的可视化、抽象问题的形象化是人们研究科学、认识世界的重要手段。

MATLAB 之所以在科技界受到人们青睐，不仅在于它的数值计算能力非凡，而且它可将计算的数据及科学问题形象地显示出来。它可以有二维、三维甚至四维的图形表现，通过对图形的线型、刻面、色彩、渲染、光线、视角等属性的处理，把计算结果表现得尽善尽美。

MATLAB 提供了许多高级绘图命令，它们极易掌握，可用来画常用的图形，如曲线图、直方图、面积图、极坐标图、等高线图、矢量场图、曲面图等等。

MATLAB 也提供了许多低级绘图命令，供有编程经验的用户，用句柄控制随心所欲地绘图，进行高级开发。

5.1 二维图形

5.1.1 基本绘图

MATLAB 有命令窗、图形窗还有程序编辑器，各自都是独立的。当人们在键入命令或运行程序画图时，系统会自动打开图形窗画图。

MATLAB 最基本、最常用的命令是 plot，其绘制二维曲线时的基本调用格式：

plot(X)

若 X 为向量，系统则以自然数列为横坐标值，以 X 元素值为纵坐标，绘制折线。若 X 为实数阵，则按列绘制每列元素值相对其下标的折线；若 X 为复数阵，则分别以 X 实部和虚部为横纵坐标绘制折线，相当于 PLOT(real(X),imag(X))。

plot(X,Y)

若 X,Y 是同维向量，则绘制以 X,Y 元素为横纵坐标的折线；若 X 是矩阵，Y 是向量，情况与上相同，只是曲线都以 Y 为共同纵坐标。若 X,Y 是同维矩阵，则以 X,Y 对应列元素为横纵坐标分别绘制曲线，曲线条数等于矩阵的行数。

plot(X1,Y1,X2,Y2,…)

在此格式中,每个二元对X-Y 的作用与plot(X,Y)相同。在以上三种格式里,输入变量X,Y 都可以是表达式,只要这表达式的运算结果符合上述格式要求。

【例 5.1.1-1】 单向量输入格式,并画连线(如图 5.1.1-1 所示)。

```
x=[1,3,2,5,4,7,6,9,8,0];
plot(x)
```

【例 5.1.1-2】 当输入参变量一个是自变量向量,另一个是应变量向量,以横纵坐标画单根曲线(如图 5.1.1-2)。

```
x=0:pi/100:2*pi;
y=sin(3*x+pi/16);
plot(x,y);
```

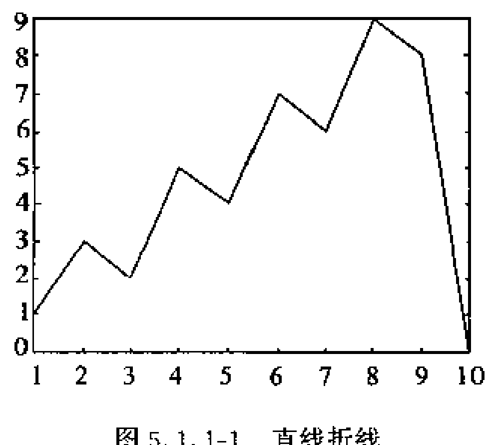


图 5.1.1-1　直线折线

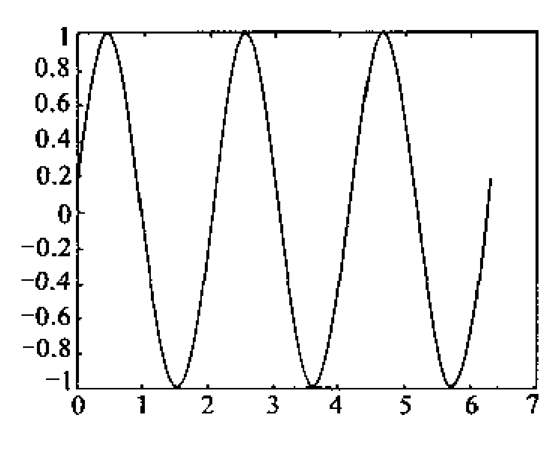


图 5.1.1-2　单根曲线

【例 5.1.1-3】 当输入参变量一个是自变量向量,另一个是应变量向量组成的矩阵,以相同横坐标画多根曲线(如图 5.1.1-3 所示)。

```
clf
x=0:pi/100:2*pi;
x=x';
y=[sin(x),sin(2*x),sin(x+pi/2),2*sin(x)]      %不同的应变量构成矩阵
plot(x,y);
%如果改用一个是自变量与一个是应变量组成向量对,结果相同
y1=sin(x),y2=sin(2*x),y3=sin(x+pi/2),y4=2*sin(x)
plot(x,y1,'r',x,y2,'g',x,y3,'b',x,y4,'m')
axis([0,6.28, -2, 2])
```

输入参变量中,符号'r','g','b','m'都是绘图控制参数,绘出不同颜色的曲线。

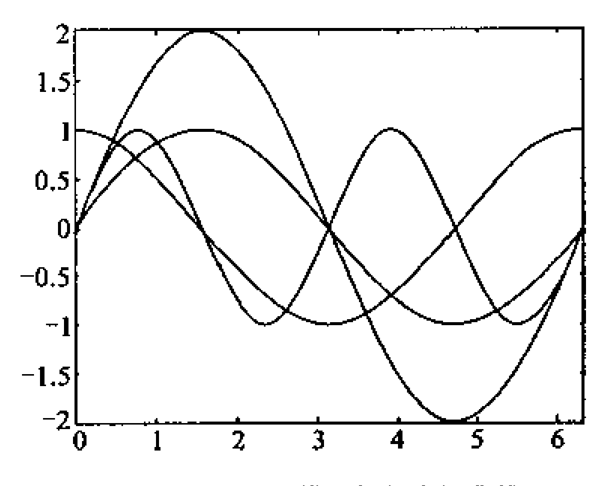


图 5.1.1-3 同一横坐标的多根曲线

5.1.2 绘图控制参数

二维绘图指令 plot 还提供一组控制曲线线型和颜色的参数。该开关总跟在横纵向量对的后面，具体如下：

颜色的控制

plot(X,S)

plot(X,Y,S)

plot(X1,Y1,S1,X2,Y2,S2,…)

其中，S 是由表 5-1 和表 5-2 中的字符组成的并带有单引号的字符串。

表 5-1 **颜色的控制符号**

颜色	RGB 值	颜色	RGB 值	颜色	RGB 值
Y	黄(1 1 0)	R	红(1 0 0)	W	白(1 1 1)
M	洋红(1 0 1)	G	绿(0 1 0)	K	黑(0 0 0)
C	青色(0 1 1)	B	蓝(0 0 1)		

线型数据标记

栅格的控制

对二维或三维图形都适用，可在图形窗口中，画出纵横平行的分格线，以便我们分析图形。

表 5-2　　曲线线型、数据标记类型

线型符号	说　明	数据显示标记	说　明
-	实线	.	小黑点
:	虚点线	o	小圈号
-.	点划线	x	叉号
--	双划线	+	十字号
		*	星号

grid on　　打开栅格绘制开关，使此后绘制的图形都带栅格。

grid off　　关闭栅格绘制开关，使此后绘制的图形都不带栅格。

grid　　实现栅格绘制与否的切换。

【例 5.1.2-1】 采用不同线型、颜色进行绘图。

```
x = 0 : 10; y = sin(x);
xi = 0 : 0.25 : 10;
yi = spline(x,y,xi);
plot(x,y,'ob',xi,yi,'-r')
```

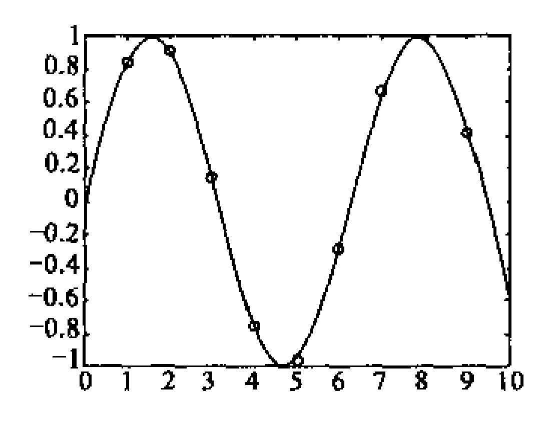


图 5.1.2-1　各种线型颜色绘图

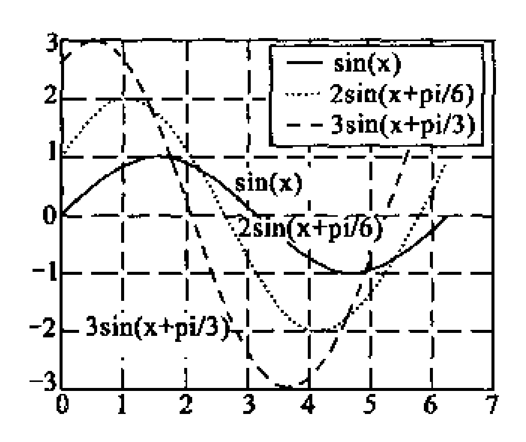


图 5.1.3-1　图形的颜色线型和标注

5.1.3　图形的标注

MATLAB 可以在画出的图形上加各种标注及文字说明。标注的内容可以是外文字符，也可以是中文字符。具体如下：

title('string')　　在图形的顶端加注文字作为图名。

xlabel('string')　　在当前图形的 x 轴旁边标注文字内容。

legend('string1', 'string2',…)　　在一个线框中,对当前图形每一子图进行标注。

text(x,y,z, 'string')　　在点(x,y,z)坐标上标注文字string。

gtext('string')　　在鼠标指定位置上标注文字string。

说明:

(1) 给y,z轴标注的命令是ylabel,zlabel,与xlabel的调用格式完全一样。

(2) 关于legend,如果用户想改变说明框的位置,可以按住鼠标把它拖到合适的地方。

(3) 在text命令中,如果x,y,z是向量,则在这三个向量定义的每一点上标注文字string。如果string是由字符串组成的字符列向量,则在每一点上标注相应的字符。

(4) 对于gtext指令,先利用鼠标定位,再在此位置标注文字,该指令不支持三维图形。

【例5.1.3-1】 采用不同线型、颜色进行绘图(如图5.1.3-1所示)。

```
clf
x=0:pi/100:2*pi;
x=x';
y1=sin(x);y2=2*sin(x+pi/6);y3=3*sin(x+pi/3);
plot(x,y1,'-r',x,y2,':g',x,y3,'-.b')
grid
hold on
legend('sin(x)','2sin(x+pi/6)','3sin(x+pi/3)')
    title('sinc 函数系列')
    xlabel('X 轴')
    ylabel('Y 轴')
    disp('请输入标注坐标')
    gtext('sin(x)')
    gtext('2sin(x+pi/6)')
    gtext('3sin(x+pi/3)')
    hold off
```

5.1.4 二维功能图形

MATLAB所提供的绘制二维图形的命令,详见表5-3。5.1.1节详细介绍了基本绘图命令plot。表中的符号函数二维曲线绘制命令ezplot已在3.7.4节作了详细介绍。下面再介绍几个常用的命令。至于它们调用格式的详细介绍请参考用

户手册或用 help 功能在线查询。

表 5-3 绘制二维图形的指令

bar	直方图	area	面积图
bar3	三维直方图	pcolor	伪彩图
contour	平面的等高线图	polar	极坐标曲线
errorbar	误差棒图	plot	直角坐标二维曲线
ezplot	符号函数二维曲线	quiver	矢量场图
feather	沿 x-轴分布的复数向量图	rose	统计频数扇形图
fplot	数值函数二维曲线	pie	饼图
fill	平面多边形填色	compass	原点出发的复数向量图
gplot	绘图论图	stem	火柴杆图
hist	统计频数直方图	stairs	阶梯曲线图

【例 5.1.4-1】 用 bar(y)和 pie(y)绘制向量 y 的直方图(如图 5.1.4-1 所示)。

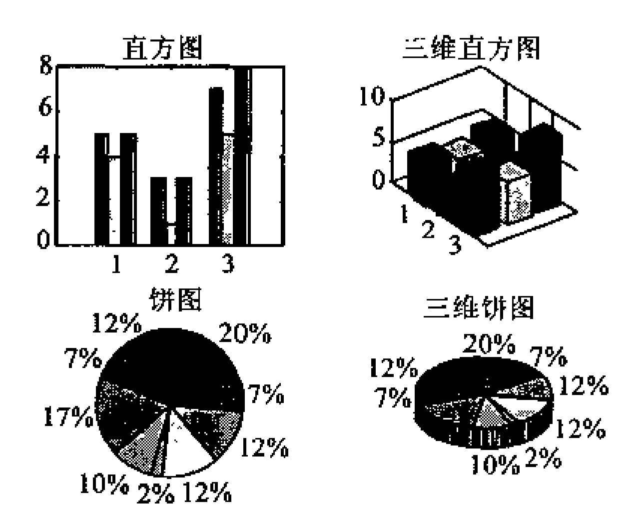


图 5.1.4-1 直方图和饼图

```
clf
rand('seed',5)
y=fix(10*rand(3));
subplot(2,2,1); bar(y);title('直方图'); %subplot 的详细解释见 5.3.2 节
```

```
  subplot(2,2,2); bar3(y);title('三维直方图');
subplot(2,2,3); pie(y);title('饼图');
  subplot(2,2,4); pie3(y);title('三维饼图');
```

【例 5.1.4-2】 用errorbar(x,y,e) 绘制误差棒图(如图 5.1.4-2 所示)。

```
x=0:pi/12:2*pi;
y=sin(x);
e=rand(size(x))/5;
errorbar(x,y,e);
axis([0,6.28,-1.5 1.5])
```

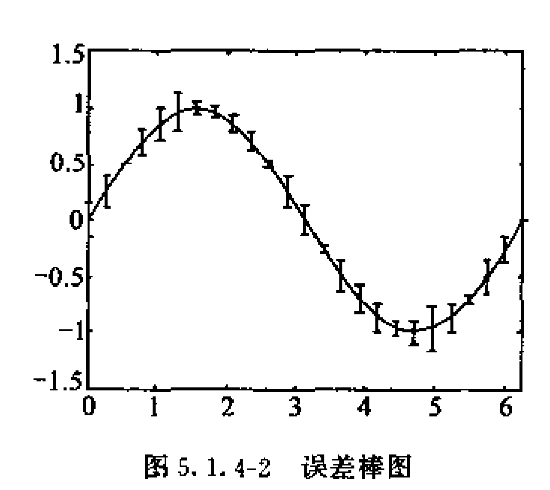


图 5.1.4-2　误差棒图

【例 5.1.4-3】 用stem(x,y)绘制火柴杆图(如图 5.1.4-3 所示)。

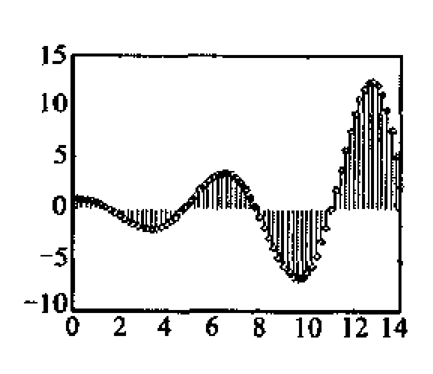


图 5.1.4-3　火柴杆图

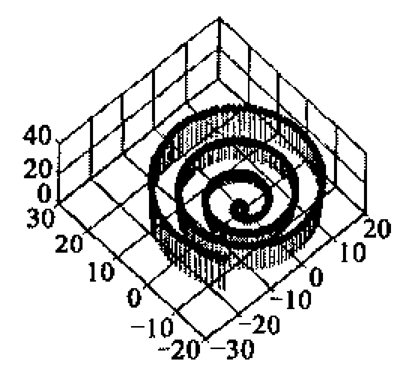
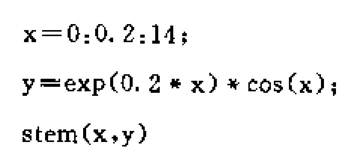


图 5.1.4-4　三维火柴杆图



```
x=0:0.2:14;
y=exp(0.2*x)*cos(x);
stem(x,y)
```

【例 5.1.4-4】 用stem3(x,y)绘制三维火柴杆图(如图 5.1.4-4 所示)。

```
T=(0:360)/100*2*pi;
x=T.*cos(T);
y=T.*sin(T);
stem3(x,y,T,'fill')
view([-10 70])
rotate3d on
```

【例 5.1.4-5】 用极坐标命令 polar(t,y)绘制螺旋线(如图 5.1.4-5 所示)。

```
t=0:0.1:4*pi;y=t;
polar(t,y)
%t,y是同维的向量或矩阵,t的单位是弧度,可以用开关控制线型和色彩
```

【例 5.1.4-6】 用stairs命令绘制阶梯曲线(如图 5.1.4-6 所示)。

```
x=0:pi/20:7;
y=exp(0.1*x).*sin(x);
stairs(x,y)
hold on
plot(x,y,'-*r')
hold off
```

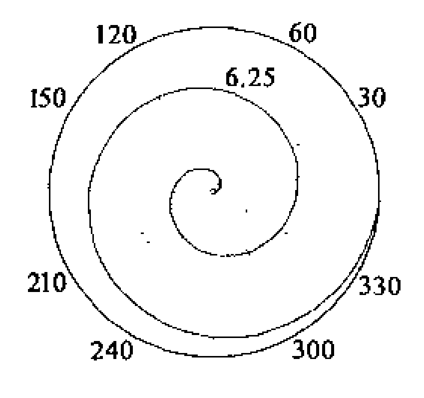


图 5.1.4-5 极坐标图

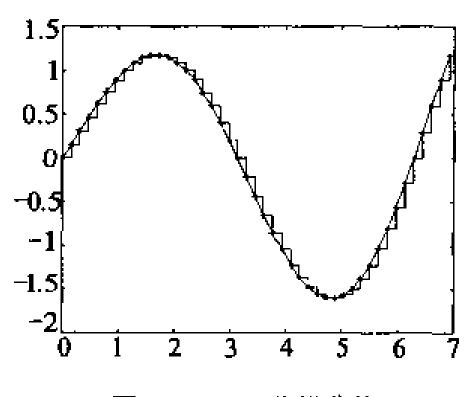


图 5.1.4-6 阶梯曲线

【例 5.1.4-7】 用area命令绘制面积图(如图 5.1.4-7 所示)。

```
rand('seed',5)
y=fix(10*rand(4));
area(y)
```

【例 5.1.4-8】 绘制矢量图。原点出发的复数向量图(如图 5.1.4-8 所示)。

```
x=pi/180*[45 90 440 440];
y=[10 7 4 8];
```

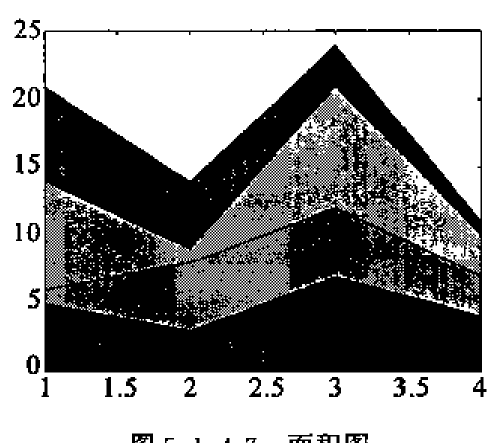


图5.1.4-7 面积图

```
[x,y]=pol2cart(x,y) %极坐标变成直角坐标数据
compass(x,y)
```

【例5.1.4-9】 沿 x 轴分布的复数向量图(如图5.1.4-9所示)。

```
T= (350:-10:10)*pi/180; r = 2*ones(size(T));
[u,v] = pol2cart(T,r);
feather(u,v), axis equal
```

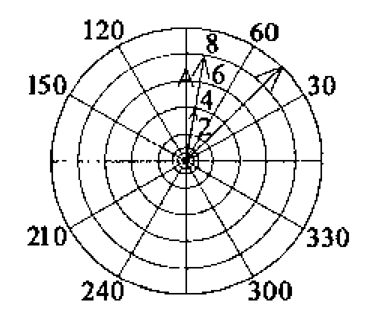


图5.1.4-8 罗盘矢量图

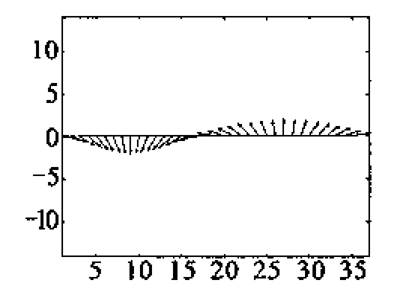


图5.1.4-9 羽毛向量图

5.2 三维图形

5.2.1 三维曲线

MATLAB提供了一个绘制三维空间曲线的基本函数plot3,许多其他的三维曲线函数都直接或间接地用到该函数。

plot3的基本调用格式是:

```
plot3(X,Y,Z)
plot3(X,Y,Z,S)
plot3(X1,Y1,Z1,S1,X2,Y2,Z2,S2,…)
```

X,Y,Z,S 与函数 plot 定义相同。

【例 5.2.1-1】 指定线型、色彩的三维曲线。

```
t=0:pi/50:10*pi;
z=t;x=sin(t);y=cos(t);
plot3(x,y,z,'b-');   % 如图 5.2.1-1 所示
```

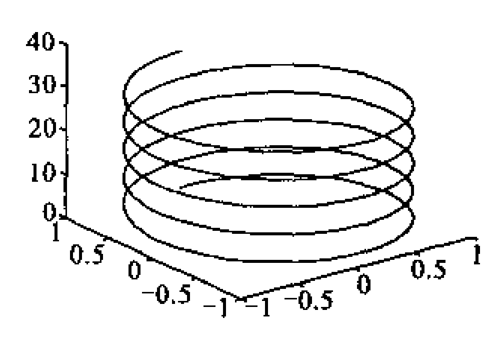


图 5.2.1-1 三维曲线

5.2.2 三维线框图

三维线框图的形成原理是：首先，对 x-y 平面用与 x,y 坐标轴平行的直线分成若干小矩形，并计算矩形网格点上的函数值，得到相应三维曲面上的数据点，将这些数据点分别用空间曲线连接，得到 *mesh* 线框图。

二元函数 $z=f(x,y)$ 图上的栅格交点满足以下关系：

$$z_{ij}=f(x_{ij},y_{ij})\quad i=1,\cdots,n;j=1,\cdots,m.$$

其中，x_{ij},y_{ij}是栅格交点的 x-y 平面坐标。

当栅格用 n 维 x 向量和 m 维 y 向量表示，则可用以下命令求得栅格交点的 x-y 坐标数组成 $\boldsymbol{X}$,$\boldsymbol{Y}$ 向量。

```
[X,Y]=meshgrid(x,y)
```

进而利用数组运算可求得相应的 **Z** 阵。

mesh 线框图调用格式

```
mesh(Z)
mesh(X,Y,Z)
```

如果 $\boldsymbol{X}$,$\boldsymbol{Y}$ 是向量，那么 $\boldsymbol{X}$ 的长度等于矩阵 $\boldsymbol{Z}$ 的列维；$\boldsymbol{Y}$ 的长度等于矩阵 $\boldsymbol{Z}$ 的行维。网格数据点的 x-y 坐标均由 $\boldsymbol{X}$,$\boldsymbol{Y}$ 向量构成，纵坐标则取自阵 $\boldsymbol{Z}$。

如果 X,Y,Z 是同维矩阵，则数据点的坐标分别取自这三个阵。

【例 5.2.2-1】 绘制 $z=x^2+y^2$ 的三维网图形(如图 5.2.2-1 所示)。

```
clf
x=-4:4;y=x
[X,Y]=meshgrid(x,y);        %生成网格点的 x-y 坐标
Z=X.^2+Y.^2;
mesh(X,Y,Z);colormap([1 0 0]);
hold on %hold on 屏幕保护详细解释见 5.6.3 节
Z0=zeros(size(x));
plot3(X,Y,Z0,'ko')       %绘制网格点在 x-y 平面上的投影
stem3(X,Y,Z,'fill')
hold off
```

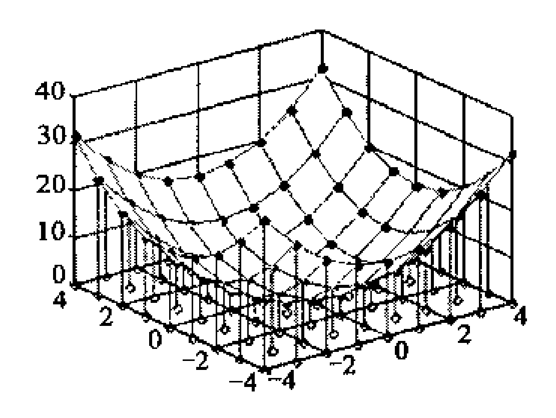


图 5.2.2-1　栅格点与三维曲面

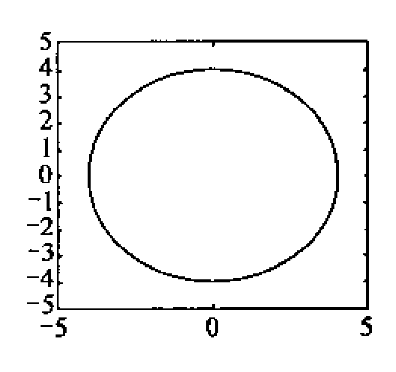


图 5.2.3-1　圆

5.2.3　基本图形元素的生成

1. 圆

【例 5.2.3-1】 圆(如图 5.2.3-1 所示)。

```
t=0:pi/50:2*pi;
x=4*cos(t);y=4*sin(t);
plot(x,y); axis('square')
axis([-5 5 -5 5]);hidden on;
```

2. 螺旋线

【例 5.2.3-2】 螺旋线(如图 5.2.1-1 所示)。

3. 平面

【例 5.2.3-3】 平面(如图 5.2.3-2 所示)。

```
0.5:8;y=x; clf;x=-8; [X,Y]=meshgrid(x,y);
Z=2 * ones(size(X));
%Z=X+Y;
% Z=X;
mesh(X,Y,Z )
```

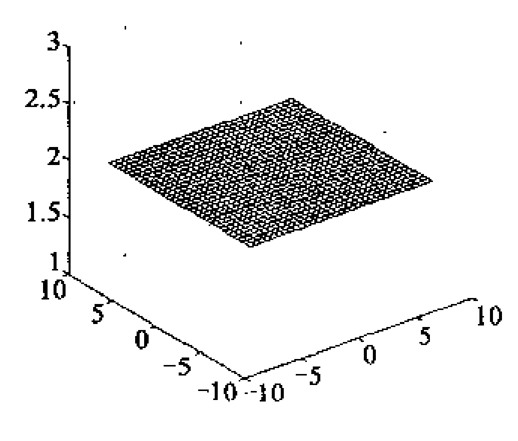


图 5.2.3-2 平面

4. 椭圆面

【例 5.2.3-4】 椭圆面(如图 5.2.2-1 所示)。

```
x=-4:4;y=x
[X,Y]=meshgrid(x,y);      %生成网格点的 x-y 坐标
Z=X.^2+Y.^2;
mesh(X,Y,Z);
```

5. 抛物面

【例 5.2.3-5】 双曲抛物面(如图 5.2.3-3 所示)。

```
a=5i;b=4i; c=4;d=1;
xgrid=linspace(-abs(a),abs(a),20); % x 网格坐标
ygrid=linspace(-abs(b),abs(b),20); % y 网格坐标
[x,y]=meshgrid(xgrid,ygrid);      %生成网格点的 x-y 坐标
z=c * sqrt(d-x. * x/a/a-y. * y/b/b);
    u=1;                          % u=1 为负半面,并加负坐标轴
    z1=real(z);
    for k=2:N-1                   %整个循环为取消 z 中的虚数的点
        for j=2:N-1
          if imag(z(k,j))~=0  z1(k,j)=0;
```

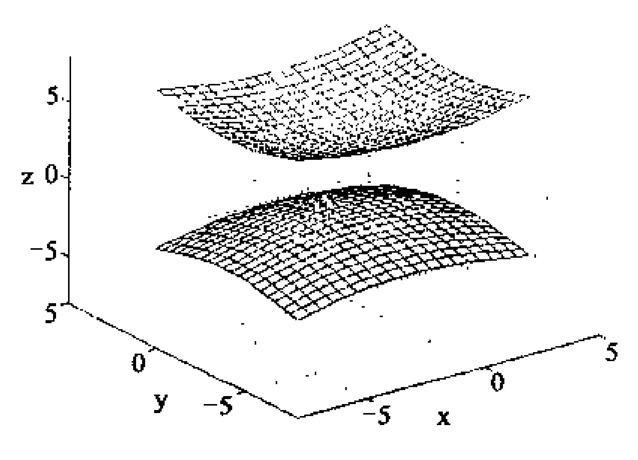


图 5.2.3-3 双曲抛物面

```
endif
all(imag(z([k−1:k+1],[j−1:j+1])))~=0      z1(k,j)=NaN;
      end
    end
  end
  mesh(x,y,z1),hold
  if u==1 z2=−z1;mesh(x,y,z2);
        axis([−abs(a),abs(a),−abs(b),abs(b),−abs(c),abs(c)]);
  end
  xlabel('x'); ylabel('y'); zlabel('z');
  axis([−8,5,−8,5,−8,8]);
  hold off;
```

6. 回转面

回转面可看成是用母线绕轴线旋转生成,它用母线向量r和旋转圆上的纬线条数n定义。于是利用下列指令可得到柱面的x-y-z坐标矩阵[X,Y,Z]:

[X,Y,Z]=cylider(r,n)

说明:

(1)"母线"可看做是在旋转轴为横坐标及旋转半径为纵坐标上定义的曲线。向量r则是旋转轴上等分刻度上定义的半径向量。

(2)由该指令所得的X,Y,Z可通过网线图命令mesh或表面指令surf表现为图形。

【例5.2.3-6】 回转曲面图(如图5.2.3-4所示)。

```
subplot(2,2,1)
```

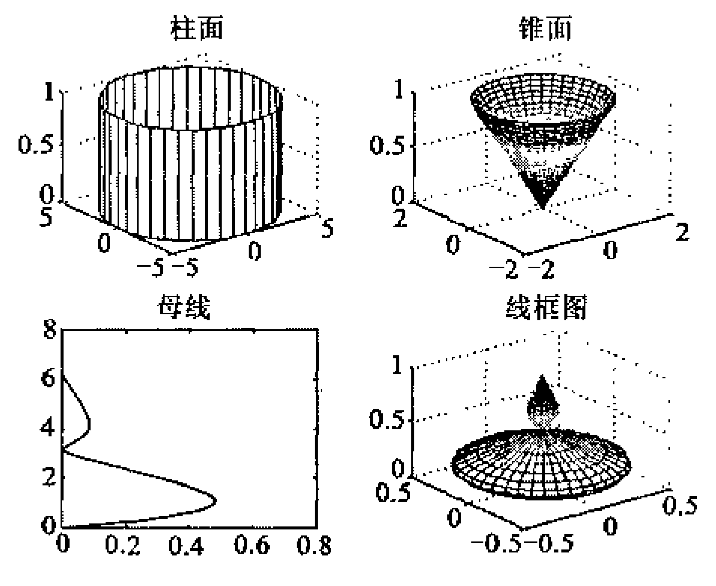


图 5.2.3-4 回转曲面

```
[X,Y,Z]=cylinder(5,40);
mesh(X,Y,Z)
title('柱面')
subplot(2,2,2)
t=0:pi/12:2*pi;
r=t/4;
[X,Y,Z]=cylinder(r,40);
mesh(X,Y,Z)
title('锥面')
t=0:pi/12:2*pi;
r=abs(exp(-0.55*t).*sin(t));
subplot(2,2,3)
plot(r,t,'r')
title('母线')
%disp('请按任意键')
%pause
subplot(2,2,4)
[X,Y,Z]=cylinder(r,40);
mesh(X,Y,Z)
title('线框图');colormap('default');
```

7. 球面

球面可看做是由一个圆绕着通过圆心的轴线旋转一周而形成的。其命令：

[X,Y,Z]=sphere(n)

指令中X,Y,Z和n的含义均与柱面相同。

【例5.2.3-7】 球面图(如图5.2.3-5所示)。

```
[X,Y,Z]=sphere(30);
mesh(X,Y,Z)
```

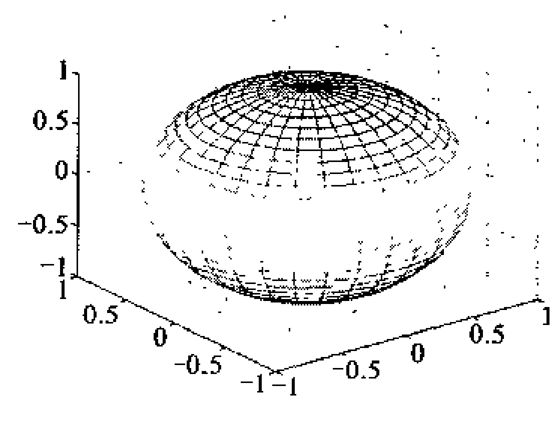


图5.2.3-5　球面

5.3 几何变换

几何变换是计算机图形学的重要章节。在图形显示中，常常需要对图形进行平移、放大、投影、透视等等，原理是将原图形矩阵 $\boldsymbol{A}$ 乘上一个变换矩阵 $\boldsymbol{T}$，将变换结果绘制出来就是变换后图形。

5.3.1 二维图形变换

二维平面上一点可以用矩阵的向量表示(矢量 $\boldsymbol{R}^2$ 空间的一个元素)。使用矩阵可以进行几何变换(从一个二维空间到另外一个二维空间)。例如，$\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}$ 是将 $\boldsymbol{x}$ 矩阵经 $\boldsymbol{A}$ 变换到平面上的另外一点或另外一个矢量。不同的矩阵具有不同的作用。它们可以进行缩放、错切、反射、旋转、平移等几何变换。

1. 比例变换

变换矩阵　$T=\begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix}$

【例5.3.1-1】 将 $\boldsymbol{A}$ 阵图形作比例放大变换，取 $a=d=2$，如图5.3.1-1所示。

```
A=[1 3 2.5 1; 1 1 2 1]'
x=A(:,1)
y=A(:,2)
line (x,y);hold on
T1=[2 0;0 2]
AA=A*T1;
x=AA(:,1)
y=AA(:,2)
line(x,y)
```

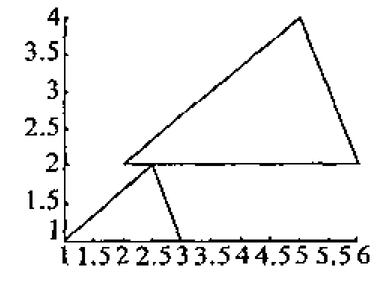


图 5.3.1-1　图形放大

2. 错切变换

变换矩阵　$T2=\begin{bmatrix}1 & b\\ c & 1\end{bmatrix}$

【例 5.3.1-2】　将 A 阵图形作错切变换，取 $c=2$，$b=0$，如图 5.3.1-2 所示。

```
A=[1 3 2.5 1; 1 1 2 1]'
x=A(:,1)
y=A(:,2)
line(x,y);hold on
T2=[1 0;2 1]
AA=A*T2;
x=AA(:,1)
y=AA(:,2)
line(x,y)
```

图 5.3.1-2　错切变换

3. 反射变换

变换矩阵　$T3=\begin{bmatrix}-1 & 0\\ 0 & 1\end{bmatrix}$

【例 5.3.1-3】　将 A 阵图形以 Y 轴作反射变换，如图 5.3.1-3 所示。

```
A=[1 3 2.5 1; 1 1 2 1]'
x=A(:,1)
y=A(:,2)
line(x,y);hold on
T3=[-1 0;0 1]
AA=A*T3;
x=AA(:,1)
```

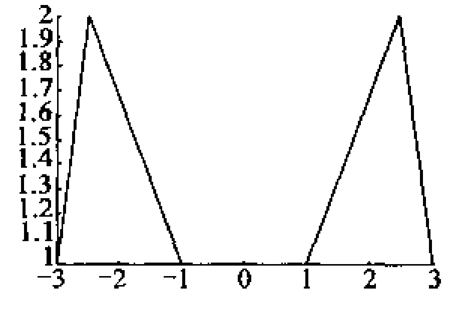


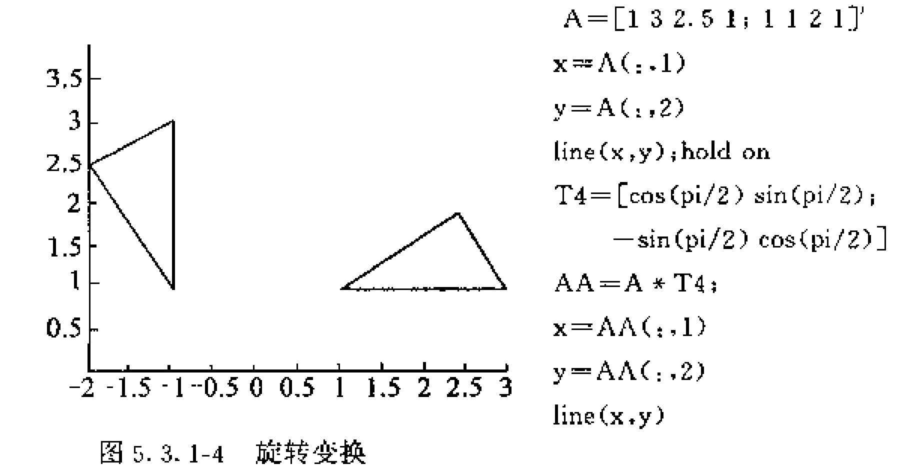
图 5.3.1-3　反射变换

```
y=AA(:,2)
line(x,y); axis('on')
```

4. 旋转变换

变换矩阵　$T4=\begin{bmatrix}\cos 90^\circ & \sin 90^\circ \\ -\sin 90^\circ & \cos 90^\circ\end{bmatrix}$，旋转角为 90°。

【例 5.3.1-4】　将 *A* 阵图形以原点作旋转变换，如图 5.3.1-4 所示。




```
A=[1 3 2.5 1; 1 1 2 1]'
x=A(:,1)
y=A(:,2)
line(x,y);hold on
T4=[cos(pi/2) sin(pi/2);
    -sin(pi/2) cos(pi/2)]
AA=A * T4;
x=AA(:,1)
y=AA(:,2)
line(x,y)
```

图 5.3.1-4　旋转变换

5. 奇次坐标-平移

变换矩阵　$T5=\begin{bmatrix}1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & l\end{bmatrix}$

【例 5.3.1-5】　将 *A* 阵作平移变换，如图 5.3.1-5 所示。

```
A=[1 3 2.5 1; 1 1 2 1; 1 1 1 1]'
x=A(:,1)
y=A(:,2)
line(x,y);hold on
T5=[1 0 0;0 1 0; 5 5 1]
AA=A * T5;
x=AA(:,1)
y=AA(:,2)
line(x,y)
```

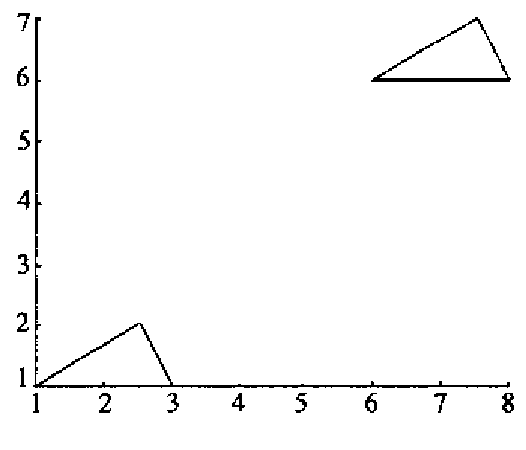


图 5.3.1-5　平移变换

5.3.2　三维图形

三维图形变换与二维图形变换相似，只是矩阵多了一列。

1. 比例变换

$$\text{变换矩阵}\quad T=\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & e & 0 & 0 \\ 0 & 0 & i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

【例 5.3.2-1】 将 *A* 阵作比例变换，如图 5.3.2-1 所示。

```
A=[12 15 25 1;0 15 25 1;0 15 0 1;35 15 0 1;12 15 25 1;...
12 0 25 1;0 0 25 1;0 0 0 1;35 0 0 1;...
35 15 0 1;12 15 25 1; 12 0 25 1;35 0 0 1;...
      0 0 0 1;0 0 25 1;0 15 25 1;0 15 0 1;0 0 0 1]
x=A(:,1)
y=A(:,2)
line(x,y,z);hold on
z=A(:,3)
T=[2 0 0 0; 0 3 0 0; 0 0 4 0; 0 0 0 1] %比例
AA=A*T;
x=AA(:,1)
y=AA(:,2)
z=AA(:,3)
line (x,y,z);hold on
rotate3d on
```

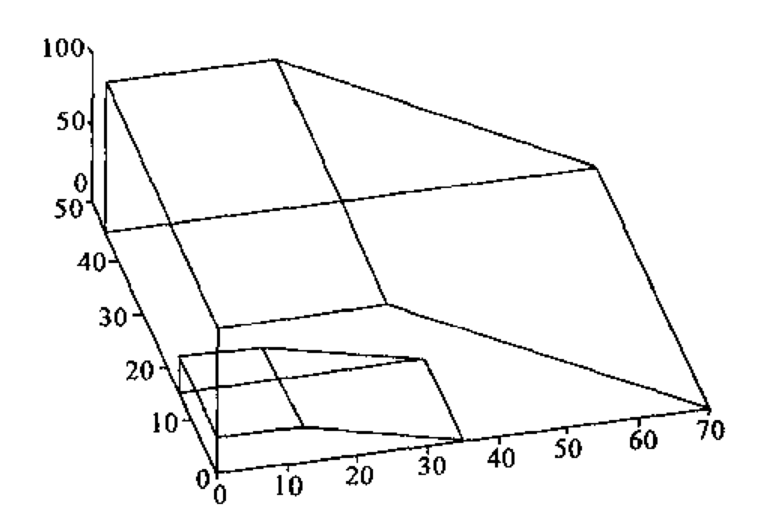


图 5.3.2-1　比例变换

2. 三视图变换

俯视图主视图变换矩阵(向 v 面投影) $\boldsymbol{T}_v=\begin{bmatrix}1&0&0&0\\0&0&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}$

变换矩阵 $\boldsymbol{T}_h=\boldsymbol{T}_{\text{向}h\text{面投影}}\cdot\boldsymbol{T}_{\text{绕}x\text{轴顺时针旋转90度}}\cdot\boldsymbol{T}_{\text{平移}-n}$　即 $\boldsymbol{T}_h=\begin{bmatrix}1&0&0&0\\0&1&-1&0\\0&0&0&0\\0&0&-n&1\end{bmatrix}$

侧视图变换矩阵 $\boldsymbol{T}_w=\boldsymbol{T}_{\text{向}w\text{面投影}}\cdot\boldsymbol{T}_{\text{绕}z\text{轴顺时针旋转90度}}\cdot\boldsymbol{T}_{\text{沿}x\text{轴方向平移}1}$，即

$$\boldsymbol{T}_w=\begin{bmatrix}0&0&0&0\\1&0&0&0\\0&0&1&0\\1&0&0&1\end{bmatrix}$$

【例 5.3.2-2】　将 A 阵图形作三视图变换，如图 5.3.2-2 所示。

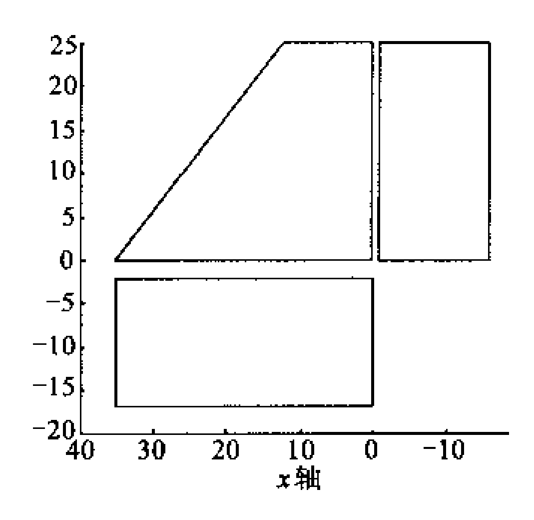


图 5.3.2-2　平移变换

```
clf
A=[12 15 25 1;0 15 25 1;0 15 0 1;35 15 0 1;12 15 25 1;...
12 0 25 1;0 0 25 1;0 0 0 1;35 0 0 1;...
    35 15 0 1;12 15 25 1; 12 0 25 1;35 0 0 1;...
            0 0 0 1;0 0 25 1;0 15 25 1;0 15 0 1;0 0 0 1];
Tv=[1 0 0 0;0 0 0 0 ;0 0 1 0; 0 0 0 1]; %主视图
Th=[1 0 0 0;0 0 -1 0 ;0 0 0 0; 0 0 -2 1]; %俯视图,向下平移-2
```

```
Tw=[0 0 0 0;-1 0 0 0 ;0 0 1 0; -1 0 0 1]; %侧视图，向右平移1
  AV=A * Tv;
x=AV(:,1);
y=AV(:,2);
z=AV(:,3);
line (x,y,z);hold on;
AH=A * Th;
x=AH(:,1);
y=AH(:,2);
z=AH(:,3);
line (x,y,z);hold on;
AW=A * Tw;
      x=AW(:,1);
y=AW(:,2);
z=AW(:,3);
line (x,y,z);
hold on;xlabel('x 轴');view([180 0])
```

3. 正轴侧变换

正轴侧变换矩阵 $T=T_{\text{绕z轴顺时针旋转}i}\cdot T_{\text{绕x轴顺时针旋转}j}\cdot T_{\text{向v面投影}}$，即

$$T=\begin{bmatrix}\cos I & 0 & \sin I\cdot\sin J & 0\\ -\sin I & 0 & \cos I\cdot\sin J & 0\\ 0 & 0 & \cos J & 0\\ 1 & 0 & 0 & 1\end{bmatrix}$$

【例 5.3.2-3】 将 A 阵作正轴侧变换，如图 5.3.2-3 所示。

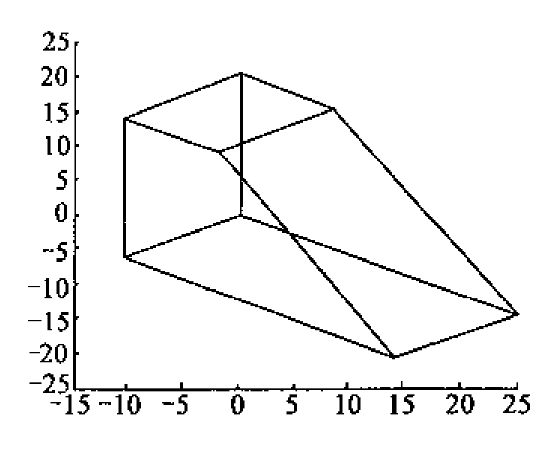


图 5.3.2-3 正轴侧变换

```
clf
A=[12 15 25 1;0 15 25 1;0 15 0 1;35 15 0 1;12 15 25 1;...
    12 0 25 1;0 0 25 1;0 0 0 1;35 0 0 1;...
    35 15 0 1;12 15 25 1; 12 0 25 1;35 0 0 1;...
    0 0 0 1;0 0 25 1;0 15 25 1;0 15 0 1;0 0 0 1];
x=A(:,1)
y=A(:,2)
z=A(:,3)
i=pi/4;j=35.5*pi/180;
T1=[cos(i) sin(i) 0 0;-sin(i) cos(i) 0 0;0 0 1 0; ...
0 0 0 1]  %绕z轴旋转
T2=[1 0 0 0;0 cos(j) -sin(j) 0;0 sin(j) cos(j) 0 ; ... 0 0 0 1]  %绕x轴旋转
T3=[1 0 0 0;0 0 0 0 ;0 0 1 0; 0 0 0 1];
T=T1*T2*T3;
AA=A*T;
x=AA(:,1);
y=AA(:,2);
z=AA(:,3);
line (x,y,z);hold on;
view([ 0 0]);
```

4. 斜轴侧变换

斜轴侧变换矩阵 $T=T_{沿X含Y错切}\cdot T_{沿Z含Y错切}\cdot T_{向v面投影}$，即 $T=\begin{bmatrix}1&0&0&0\\d&0&f&0\\0&0&0&0\\0&0&0&1\end{bmatrix}$。

【例 5.3.2-4】 将 A 阵作斜轴侧变换，如图 5.3.2-4 所示。

```
clf
A=[12 15 25 1;0 15 25 1;0 15 0 1;35 15 0 1;12 15 25 1;...
    12 0 25 1;0 0 25 1;0 0 0 1;35 0 0 1;...
    35 15 0 1;12 15 25 1; 12 0 25 1;35 0 0 1;...
    0 0 0 1;0 0 25 1;0 15 25 1;0 15 0 1;0 0 0 1];
x=A(:,1)
y=A(:,2)
z=A(:,3)
% line(x,y,z);
f=-0.354;d=0.354;      %斜二
T1=[1 0 0 0;d 1 0 0 ;0 0 1 0; 0 0 0 1];  %沿x含y错切
```

```
T2=[1 0 0 0;0 1 f 0 ;0 0 1 0; 0 0 0 1];  %沿z含y错切
T=T1 * T2;
AA=A * T;
x=AA(:,1);
y=AA(:,2);
z=AA(:,3);
line (x,y,z);hold on;view([ 0 0]);
```

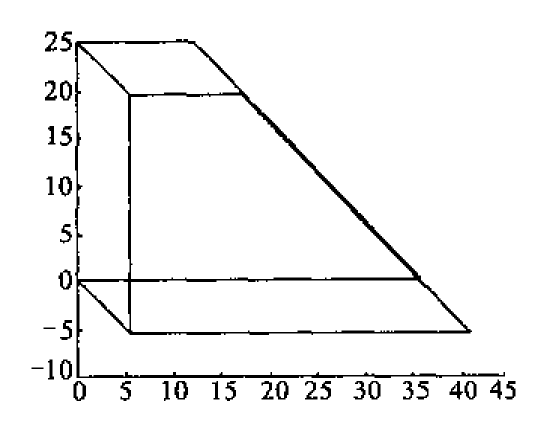


图 5.3.2-4 斜轴侧变换

5. 三点透视图变换

正轴侧变换矩阵 $T=T_{平移变换}\cdot T_{透视变换}\cdot T_{绕z轴转\theta}\cdot T_{绕x轴转\phi}\cdot T_{向V面投影}$，即

$$T=\begin{bmatrix} \cos T & 0 & \sin T\cdot\sin F & p \\ -\sin T & 0 & \cos T\cdot\sin F & q \\ 0 & 0 & \cos F & r \\ l\cos T-m\sin T & 0 & \sin F(l\sin T+m\cos T)+n\cos F & lp+mq+nr+1 \end{bmatrix}$$

【例 5.3.2-5】 将 *A* 阵作三点透视变换，如图 5.3.2-5 所示。

```
clf
a=[12 15 25 1;0 15 25 1;0 15 0 1;35 15 0 1;12 15 25 1;...
   12 0 25 1;0 0 25 1;0 0 0 1;35 0 0 1;...
   35 15 0 1;12 15 25 1; 12 0 25 1;35 0 0 1;...
   0 0 0 1;0 0 25 1;0 15 25 1;0 15 0 1;0 0 0 1];
%三点透视
   l=-1;m=-1;n=-1.5; p=-0.6;q=-0.6;r=0.65;t=30 * pi/180;  f=30
 * pi/180
T1=[1 0 0 0;0 1 0 0 ;0 0 1 0; l m n 1]; %平移
T2=[1 0 0 p;0 1 0 q ;0 0 1 r; 0 0 0 1]; %透视
```

```
%绕z轴旋转t°
T3=[cos(t) sin(t) 0 0;-sin(t) cos(t) 0 0;0 0 1 0; ...
0 0 0 1];
%绕x轴旋转f°
T4=[1 0 0 0;0 cos(f) sin(f) 0;0 -sin(f) cos(f) 0; ...
0 0 0 1];
T5=[1 0 0 0;0 0 0 0 ;0 0 1 0; 0 0 0 1]; %向v投影
T=T1 * T2 * T3 * T4 * T5;
A=a * T;
x=A(:,1);
y=A(:,2);
z=A(:,3);
line (x,y,z);hold on;view([ 0 0]);
rotate3d on
```

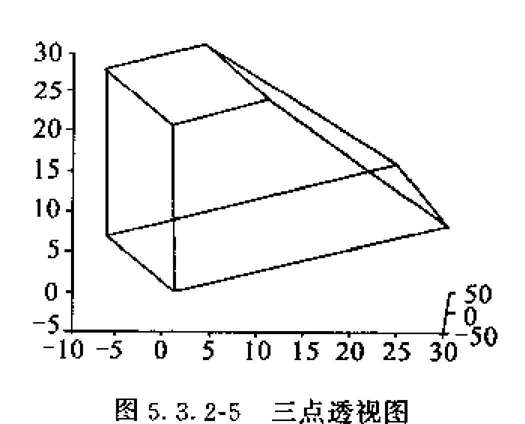


图 5.3.2-5 三点透视图

5.4 微分几何典型例子

【例 5.4-1】 求螺旋线切线、法平面、副法线、密切平面、主法线和从切平面。

一、绘出曲线

```
clf;t=0:pi/10:4 * pi;
x=cos(t);y=sin(t);
plot3(cos(t),sin(t),t);rotate3d on;
axis('square');grid on;hold on;hidden on;
t=pi * 5/2;x=cos(t);y=sin(t);z=t; plot3(x,y,z,'r * ')
```

二、求出各阶导数及其值

```
syms t
r='[cos(t) sin(t) t ]';
dr=diff(r);
ddr=diff(r,2);
ra=subs(r,pi * 5/2,'t');
dra=subs(dr,pi * 5/2,'t');
ddra=subs(ddr,pi * 5/2,'t');
crs _ ddra=cross(dra,ddra);
dr =
    [-sin(t), cos(t), 1]
ddr =
    [-cos(t), -sin(t), 0]
ra =
    [0, 1, 5 * pi/2]
dra =
    [-1, 0, 1]
    ddra =
    [ 0, -1, 0]
    crs _ ddra =
    [ 1, 0, 1]
```

三、求出切线、法平面、副法线、密切平面、主法线和从切平面的方程

$t=5\pi/2$ 处的切线方程 $\frac{x-0}{-1}=\frac{y-1}{0}=\frac{(z-5\pi/2)}{1}$.

$t=5\pi/2$ 处的主法线方程 $\frac{x-0}{0}=\frac{y-1}{-1}=\frac{z-5\pi/2}{0}$.

$t=5\pi/2$ 处的法平面方程 $(x-0)\cdot(-1)+(y-1)\cdot 0+(z-5\pi/2)\cdot 1=0$ 即 $z=x+5\pi/2$.

$t=5\pi/2$ 处的密切平面方程 $\begin{vmatrix} x-0 & y-1 & z-5\pi/2 \\ -1 & 0 & 1 \\ 0 & -1 & 0 \end{vmatrix}=0$，即 $z=5\pi/2-x$.

$t=5\pi/2$ 处的副法线方程 $\frac{x-0}{1}=\frac{y-1}{0}=\frac{(z-5\pi/2)}{1}$.

$t=5\pi/2$ 处的从切平面方程 $(x-0)\cdot 0+(y-1)\cdot(-1)+(z-5\pi/2)\cdot 0=0$ 即 $y=1$.

四、绘出切线、主法线、副法线

1. 绘出"$t=5\pi/2$"处的切线（如图 5.4-1 所示）

```
clf;t=0:pi/10:4 * pi;
```

```
plot3(cos(t),sin(t),t);rotate3d on;
axis('square');grid on;hold on;hidden on;
t=pi*5/2;x=cos(t);y=sin(t);z=t; plot3(x,y,z,'r*')
x=[ra(1) ra(1)-dra(1)];
y=[ra(2) ra(2)-dra(2)];z=[ra(3) ra(3)-dra(3)];
x=numeric(x); y=numeric(y); z=numeric(z);
plot3(x,y,z,'r-')
text(1,1,7,'\leftarrow 切线')
```

2. 绘出“$t=5\pi/2$”处的主法线

```
x=[ra(1) ddra(1)- ra(1)];
y=[ra(2) ddra(2)-ra(2)];z=[ra(3) ddra(3)- ra(3)];
x=numeric(x); y=numeric(y); z=numeric(z);
plot3(x,y,z,'g-')
text(-0.2,0.4,8,'\leftarrow 主法线')
```

3. 绘出“$t=5\pi/2$”处的副法线

```
x=[ra(1) ra(1)-crs_ddra(1)];y=[ra(2) ra(2)-crs_ddra(2)];...
z=[ra(3) ra(3)-crs_ddra(3)];
x=numeric(x); y=numeric(y); z=numeric(z);
plot3(x,y,z,'m-'); axis([-1 1 -1 1 0 15])
text(-1,1,6,7,'\leftarrow 副法线')
```

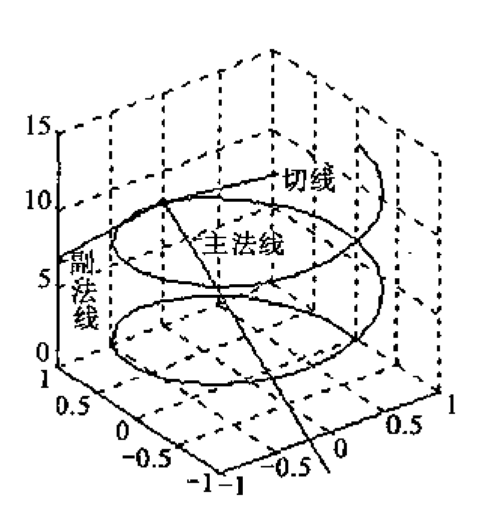


图 5.4-1 切线、主法线、副法线

五、绘出法平面、密切平面和从切平面(图 5.4-2 所示)

1. 绘出“$t=5\pi/2$”处的法平面

```
[x,y]=meshgrid(-1:0.5:1);z=x+5*pi/2;
```

```
mesh(x,y,z); axis([-1 1 -1 1 7 9]);
text(0.12,-3.2,10.75,'\leftarrow 法平面');
```

2. 绘出“$t=5\pi/2$”处的密切平面

```
[x,y]=meshgrid(-1:0.5:1);z=-x+5*pi/2;
mesh(x,y,z); text(0.2,-3,8.5,'\leftarrow 密切平面')
```

3. 绘出“$t=5\pi/2$”处的从切平面

```
[x,y]=meshgrid(-1:0.5:9);z=y;y=1*ones(size(x));
mesh(x,y,z);xlabel('x');axis([-2 1 -4 -1 8 11]);
text(-2.15,-3.5,9.75,'\rightarrow 从切平面')
```

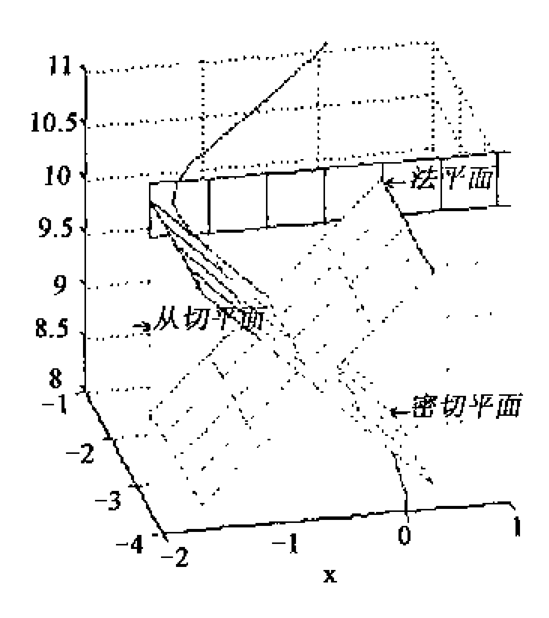


图 5.4-2 密切平面、主法线和从切平面

【例 5.4-2】 绘制曲面的切平面和法线(图 5.4-3 所示)。

(1)画出曲面

```
x=-4:0.5:4;y=x;
[X,Y]=meshgrid(x,y); %生成网格点的 x-y 坐标
Z=X.^2+Y.^2+10;
mesh(X,Y,Z);hold on;
```

(2) 求出位置矢量、导数及其值

```
syms X,Y,Z;
Z='[X^2+Y^2]+10';
r='[X Y X^2+Y^2+10]';
drx=diff(Z,'X'); dry=diff(Z,'Y');
```

```
    r0=subs(subs(r,2,'X'),-2,'Y');
    drx0=subs(drx,2,'X');dry0=subs(dry,-2,'Y');
    plot3(2,-2,18,'r*');
r0 =
    [ 2, -2, 18]
drx0 =
    4
dry0 =
-4
```

(3) 求切平面和法线方程

切平面方程 $Z-18=4(X-2)-4(Y+2)$.

法线方程 $\frac{X-2}{4}=\frac{Y+2}{-4}=\frac{Z-18}{-1}$.

(4) 绘出切平面和法线

a. 画出（2,-2,18）处的切平面

```
    x=-4:0.5:4;y=x;
    [X,Y]=meshgrid(x,y);
    Z=4*(X-2)-4*(Y+2)+18;
    mesh(X,Y,Z); % axis([-1 1 -1 1 17 19])
```

b. 画出(2,-2,18)处的法线

```
    x=[r0(1) r0(1)-drx0];
    y=[r0(2) r0(2)-dry0];
    z=[r0(3) r0(3)+1];
    x=numeric(x); y=numeric(y); z=numeric(z);
    plot3(x,y,z,'m-');axis([-4 4 -4 4 14 22]);
```

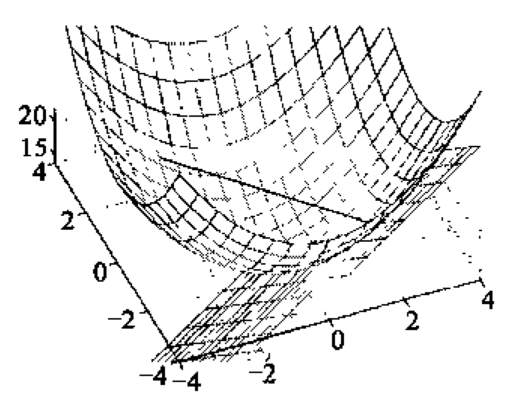


图 5.4-3　密切平面、主法线和从切平面

5.5 图形元素的相交例子

【例 5.5-1】 求两直线的交点。

直线 1：$(Y1(1)-Y1(2))X+(X1(2)-X1(1))Y-(Y1(2)X1(1)-Y1(1)X1(2))=0$

直线 2：$(Y2(1)-Y2(2))X+(X2(2)-X2(1))Y-(Y1(2)X1(1)-Y1(1)X1(2))=0$

求交点：$S(X,Y)$

(1) 编写程序

```
function [X,Y]=pll(X1,Y1,X2,Y2)
A1=Y1(1)-Y1(2);
  B1=X1(2)-X1(1);
C1= Y1(2) * X1(1)-Y1(1) * X1(2);
A2=Y2(1)-Y2(2);
  B2=X2(2)-X2(1);
C2= Y2(2) * X2(1)-Y2(1) * X2(2);
D=det([A1 B1;A2 B2]);
X=det([-C1 B1;- C2 B2])/D;
Y=det([A1 -C1;A2 - C2])/D;
```

(2) 调用程序求解

```
    x1=[1 5];y1=[1 5];x2=[1 5];y2=[5 1];
    [x,y]=Pll(x1,y1,x2,y2);
X =
    3
Y =
    3
```

【例 5.5-2】 直线与多根直线相交(如图 5.5-1 所示)。

```
    clf; xi=[1 2 3 4 5];yi=[2 6 3 6 1];
    plot(xi,yi);hold on;
X1=[1 5 ];Y1=[4 5]; line(X1,Y1);
X=zeros(size(xi));Y=X;
for i=1:5-1
X2=xi([i i+1]); Y2=yi([i i+1]);
[X,Y]=pll(X1,Y1,X2,Y2);
plot(X,Y,'r.')
```

```
end
hold off
```

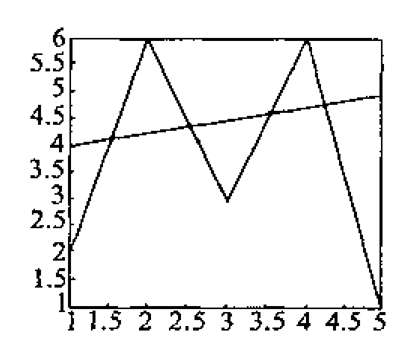


图 5.5-1　直线与多根直线相交

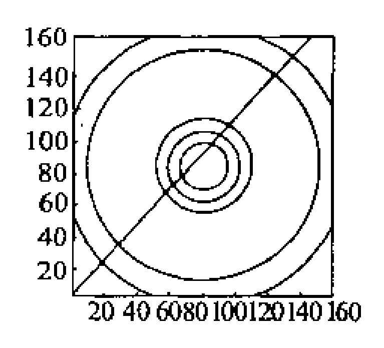


图 5.5-2　直线与曲线相交

【例 5.5-3】　直线与曲线相交(如图 5.5-2 所示)。

```
clf;x=-8:0.1:8;y=x;[X,Y]=meshgrid(x,y);
    R=sqrt(X.^2+Y.^2)+eps;Z=sin(R)./R;
contour(Z,3); hold on
c=contourc(Z,3); %计算等高线坐标数据
x=[0 360];y=[0 400];
y=(y(2)-y(1))/(x(2)-x(1))*(x-x(1))+y(1);z=[0 0];
line(x,y,z);
      c=c';
X=c(:,1);Y=c(:,2);
r0=abs(Y-(y(2)-y(1))/(x(2)-x(1))*(X-x(1))+y(1))<=.93;
%xy 小于 ε 的 0-1 矩阵
    zz=0;yy=r0.*Y;xx=r0.*X; %用 0-1 矩阵求交线上坐标数据
plot(xx(r0~=0),yy(r0~=0),'r.');
```

【例 5.5-4】　求曲线与曲线相交(如图 5.5-3 所示)。

```
clf;x=0:pi/400:2*pi;x=x';
y1=sin(pi*x);y2=cos(pi*x);plot(x,y1,x,y2);hold on
r0=abs(y2-sin(pi*x))<=0.02;   %x y 小于 ε 的 0-1 矩阵
yy=r0.*y1;xx=r0.*x;   %用 0-1 矩阵求交线上坐标数据
plot(xx(r0~=0),yy(r0~=0),'r.');   %绘出交线
rotate3d on
```

【例 5.5-5】　直线与曲面相交(如图 5.5-4 所示)。

```
clf;x=-8:0.3:8;y=x;[X,Y]=meshgrid(x,y);Z=X.^2+Y.^2;
mesh(X,Y,Z);hold on;
```

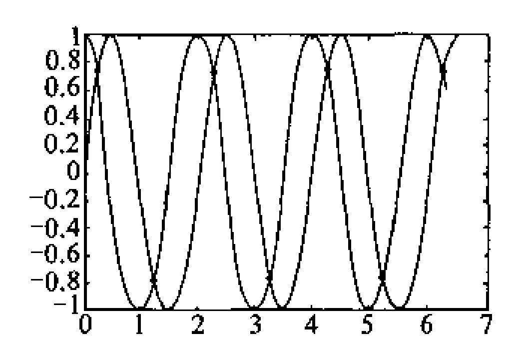


图 5.5-3 曲线与曲线相交

```
x=[-10 10];y=[-10 3]; z=[30 35]; line(x,y,z);
r0=
(abs(Y-y(1)-(y(2)-y(1))/(x(2)-x(1)) * (X-x1)))<=0.45 )&...
(abs(Z-z(1)-(z(2)-z(1))/(x(2)-x(1)) * (X-(1)))<=0.45)&...
(abs(Y-y(1)-(y(2)-y(1))/(z(2)-z(1)) * (Z-z(1)))<=0.45 );
zz=r0. * Z;yy=r0. * Y;xx=r0. * X; %用 0-1 矩阵求交线上坐标数据
plot3(xx(r0~=0),yy(r0~=0),zz(r0~=0),'r *'); %绘出交线
rotate3d on
```

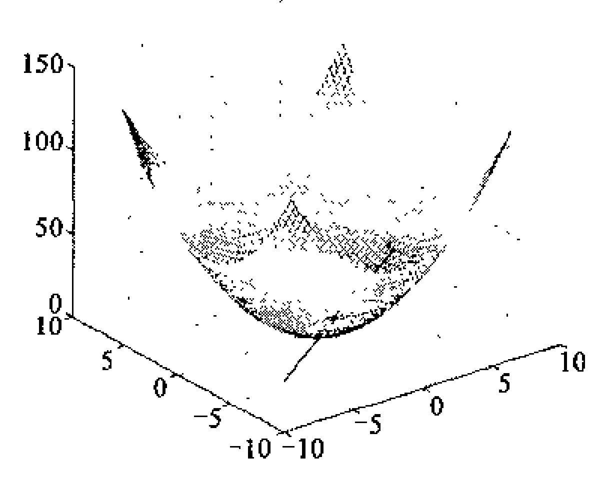


图 5.5-4 直线与曲面相交

【例 5.5-6】 平面与曲面相交(如图 5.5-5 所示)。

(1)曲面与单一平面的交线

```
clf;x=-8:0.1:8;y=x;[X,Y]=meshgrid(x,y);
Z1=2 * ones(size(X));
Z2=X.^2-Y.^2;
```

```
mesh(X,Y,Z1);hold on; mesh(X,Y,Z2 )
r0=(abs(Z1 -Z2)<=.65);   %求高差小于ε的 0-1 矩阵
zz=r0.*Z1;yy=r0.*Y;xx=r0.*X;   %用 0-1 矩阵求交线上坐标数据
  plot3(xx(r0~=0),yy(r0~=0),zz(r0~=0),'k*');   %绘出交线 colormap
(hsv) hold off;
clc
disp('观察曲面后,按任意键画交线');
pause
clf
 plot3(xx(r0~=0),yy(r0~=0),zz(r0~=0),'k.');
```

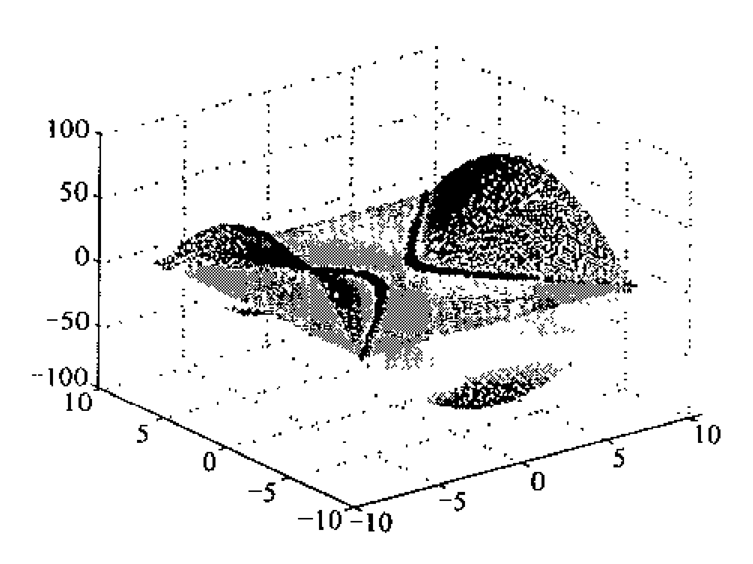


图 5.5-5 平面与曲面相交

(2)曲面与多个截平面相交(如图 5.5-6 所示)

```
clf;y=-10:0.5:10;z=y;[Z,Y]=meshgrid(z,y);
  X=Z;
X1=0*ones(size(Z));
X2=3*ones(size(Z));
X3=-3*ones(size(Z));
Z4=(X.^2-Y.^2)/10;
mesh(X1,Y,Z);hold on;
mesh(X2,Y,Z); mesh(X3,Y,Z);mesh(X,Y,Z4 )
r1= (abs(X1-X)<=0.05);
r2= (abs(X2-X)<=0.05);
r3= (abs(X3-X)<=0.05);
```

```
zz1=r1.*Z4;yy1=r1.*Y;xx1=r1.*X;   %用0-1矩阵求交线上坐标
zz2=r2.*Z4;yy2=r2.*Y;xx2=r2.*X;   %用0-1矩阵求交线上坐标
zz3=r3.*Z4;yy3=r3.*Y;xx3=r3.*X;   %用0-1矩阵求交线上坐标
%绘出交线
plot3(xx1(r1~=0),yy1(r1~=0),zz1(r1~=0),'k*');
plot3(xx2(r2~=0),yy2(r2~=0),zz2(r2~=0),'k*');
plot3(xx3(r3~=0),yy3(r3~=0),zz3(r3~=0),'k*');
colormap(hsv);
clc;disp('观察曲面后,按任意键画交线');
hold off;view([-30 45])
```

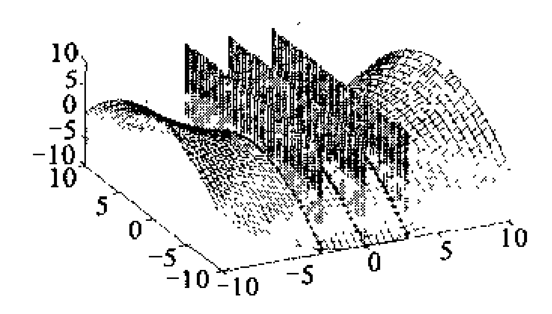


图 5.5-6 曲面与多个截平面相交

(3)曲面、水平面和侧平面的交线(如图 5.5-7 所示)。

```
clf;y=-8:0.4:8;z=y;[Z,Y]=meshgrid(z,y);
      X=Z;
      X1=0*ones(size(Z));
      Z2=0*ones(size(X));
      Z3=(X.^2-Y.^2)/10;
      mesh(X1,Y,Z);hold on; mesh(X,Y,Z2); mesh(X,Y,Z3)
r1=(abs(X1-X)<=0.05);
r2=(abs(Z3-Z2)<=0.05);
r3=(abs(X1-X)<=0.05)&(abs(Z-Z2)<=0.05);
zz1=r1.*Z3;yy1=r1.*Y;xx1=r1.*X;   %用0-1矩阵求交线上坐标
zz2=r2.*Z3;yy2=r2.*Y;xx2=r2.*X;   %用0-1矩阵求交线上坐标
zz3=r3.*Z;yy3=r3.*Y;xx3=r3.*X1;   %用0-1矩阵求交线上坐标
%绘出交线
plot3(xx1(r1~=0),yy1(r1~=0),zz1(r1~=0),'k*');
plot3(xx2(r2~=0),yy2(r2~=0),zz2(r2~=0),'k*');
plot3(xx3(r3~=0),yy3(r3~=0),zz3(r3~=0),'k*');
```

```
colormap(hsv);
view([ -30 45])
```

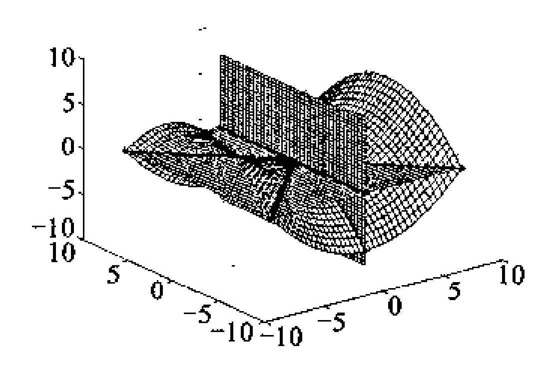


图 5.5-7　曲面、水平面和侧平面的交线

【例 5.5-7】 求曲面与曲面交线(如图 5.5-8 所示)。

```
[x,y]=meshgrid(-2:.1:2);
z1=x.*x-2*y.*y;            %绘制双曲抛物面
z2=x.*x+y.*y-5;            %绘制椭圆面
mesh(x,y,z1);hold on; mesh(x,y,z2);% shading interp ;
r0=(abs(z1-z2)<=.1);   %求高差小于 ε 的 0-1 矩阵
zz=r0.*z1;yy=r0.*y;xx=r0.*x;   %用 0-1 矩阵求交线上坐标数据
plot3(xx(r0~=0),yy(r0~=0),zz(r0~=0),'k*');   %绘出交线
colormap(cool),hold off;view(-60,30)
```

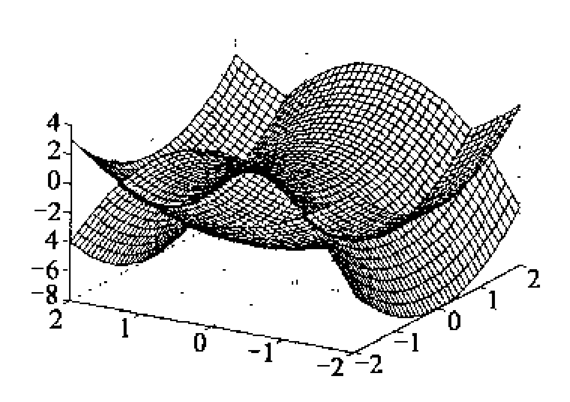


图 5.5-8　曲面与曲面交线

5.6 图形窗口

因为图形是在窗口上生成的，本节将要介绍怎样去控制窗口以实现对图形显示和操作。由于MATLAB是在win9x环境下的一个软件，提供了一些命令通过窗口来对图形进行显示和控制，这样用户在工作时就会感到很方便。

5.6.1 图形的窗口生成

通常，每当在MATLAB命令窗中的第一个绘图指令运行后，就自动创建一个名为"Figure No. 1"的图形窗口，并把图形画在这个窗口上。此后，它就被作为第一窗口。那么，所画的图形都将自动地出现在这个图形窗口上，根据命令，或把先前的图形覆盖掉，或叠加在先前的图形上。如果要绘新的图形而保留旧图形就要生成新的窗口，这样就需要使用命令来对图形窗口进行操作。其命令的调用格式如下：

Figure	打开一个新的图形窗口
Figure(n)	生成或打开第n个图形窗口，并使之成为当前窗口

5.6.2 子图形的生成和控制

在一个图形窗口(Figure)里，可以画多个有独立坐标系统的子图形，指令如下：

subplot(m,n,p)　将图形窗划分成m×n个行和列子图，并选择第p个子图为当前图形。子窗口的序号p是按行由上往下、按列自左至右递增编号。

5.6.3 图形的保护

如果用户想在某图形窗口中同时绘制不同的图形对象，方法是使用命令hold对先前图形进行保护。该指令有下面三种调用格式：

hold on　　保留当前图形，使后一图形画在当前图形上。

hold off　　返回MATLAB的缺省状态，"抹掉"当前窗中的旧图形，然后画上新图形。

hold　　在上述两个状态间切换。

图形保护的使用举例参见【例5.2.2-1】。

5.6.4 坐标轴的控制

在一般缺省情况下，MATLAB 图形坐标轴的状态是：自动显示坐标轴及刻度并采用直线坐标系。用户不需要对坐标进行任何干预，坐标刻度范围将根据绘图指令中向量或矩阵中的元素自动决定。在大部分情况下，这种缺省状态绘图是很方便的，因为图形是多种多样的。为了更有效地表现出图形特征，MATLAB 有很多关于坐标操作的 axis 命令，其调用格式如下：

axis([xmin,xmax, ymin,ymax])	指定 x-轴和 y-轴的刻度
axis([xmin,xmax,ymin,ymax,zmin,zmax])	指定三个坐标轴的刻度
axis('auto')	返回坐标轴的缺省状态、自动模式
axis('manual')	保持刻度范围不变
axis('ij')	以"矩阵(ij)"坐标轴表现
axis('xy')	使坐标轴回到(缺省状态)笛卡尔坐标系
axis('off'')	使坐标轴消失
axis('on')	打开坐标轴显示状态
axis('equal')	使各坐标轴刻度增量相同
axis('square')	使各坐标轴长度相同
axis('normal')	使坐标轴变为最大状态并且上两个命令失效

说明：

当前图形窗口坐标轴的工作状态可用以下命令进行查询：

[S1,S2,S3]=axis('state')

S1　若是 auto，则坐标轴自动确定刻度；若是 manual，则坐标轴人工确定刻度。

S2　若是 on，则显示坐标轴；若是 off，则隐藏坐标轴。

S3　若是 xy，则采用笛卡尔坐标；若是 ij，则采用"矩阵"坐标。

笛卡尔坐标是指：坐标系的原点位于左下角；x 轴水平放置，自左至右刻度；y 轴垂直设置，由下往上刻度。

矩阵坐标是指：坐标系的原点位于左上角；i 轴垂直设置，由上而下刻度；j 轴水平放置，由左到右刻度。

5.6.5 视角的设置

MATLAB 允许用户通过观察函数 view 定义所需的观察点。函数最常用的格式是：

view (az,el) 在球坐标中设置观察点 A，az 是方位角∠xoa ，el 是俯视角∠aoA。如图 5.6.5-1 所示。角度的单位是"度"。进行二维观察时，命令 VIEW(2)的缺省值为：az=0，el=90；进行三维观察时，命令 VIEW(3)的缺省值为：az=−37.5，el=30。

观察点也可以用直角坐标形式指定，格式为：view([x,y,z])。

观察点还可以直接用观察变换矩阵 T 设定，格式为：view(T)。

当前观察点位置或变换矩阵可用以下指令获得：

[az,el]=view　　返回当前观察方位角和俯视角。

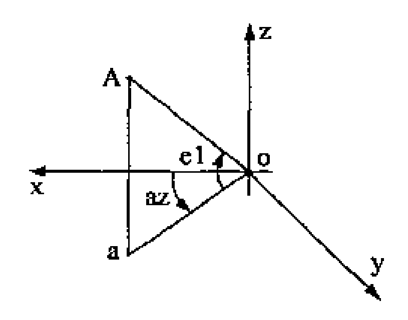


图 5.6.5-1 视角

【例 5.6.5-1】 一个曲面的四种不同视图(图 5.6.5-2)。

```
xx=[-2:0.125:2]';yy=[-1:0.125:3]';
[x,y]=meshgrid(xx',yy');z=100.*(y-x.*x).^2+(1-x).^2;
subplot(2,2,1),mesh(z),view(0,0),axis('ij'),
    title('前视图:方位角=0 俯角=0');
subplot(2,2,2),mesh(z),view(-90,0),
    title('左视图:方位角=-90 俯角=0');
subplot(2,2,3),mesh(z),view(0,90);
    title('俯视图二维缺省:方位角=0 俯角=90');
subplot(2,2,4);mesh(z),view(-37.5,30),
    title('三维缺省:方位角=-37.5 俯角=30');
```

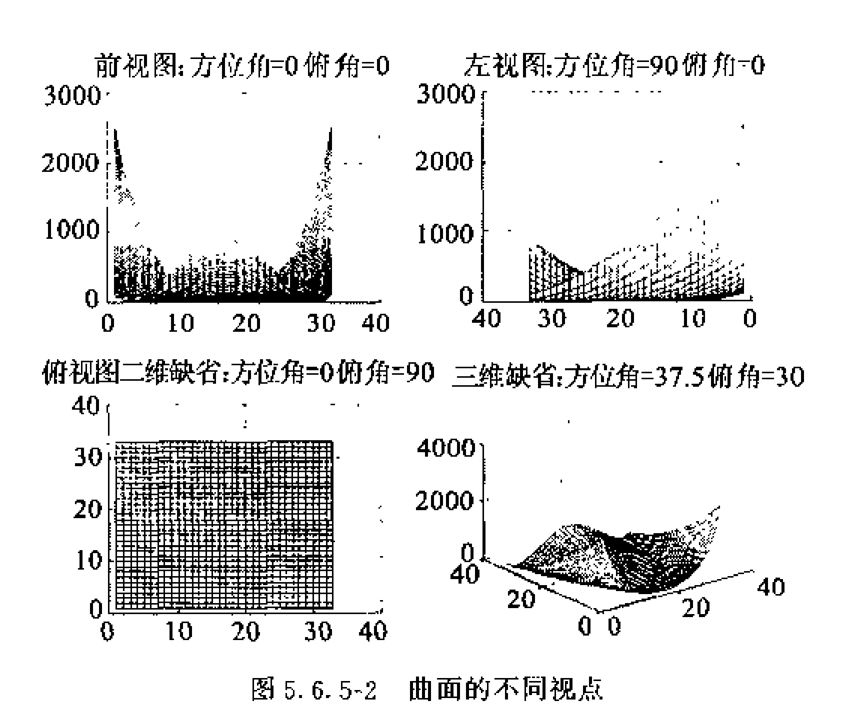


图5.6.5-2 曲面的不同视点

5.7 图形着色

5.7.1 多边形的填色

填色是指将多边形的内部填满指定的颜色。MATLAB提供两个填色函数:fill和fill3。前者用于平面多边形的填色,后者用于空间多边形的填充。多边形可以是凹的,也可以是凸的,并允许多边形的边自相交。如果填色时,为每个点指定颜色,MATLAB将采用双线性插值决定其内部表面的颜色。

二维多边形填充

二维填色命令的调用格式如下:

fill(x,y,c)

fill(x1,y1,c1,x2,y2,c2,…)

说明:

x,y(或x1,y1等)是同维向量。按向量元素下标渐增次序依次用直线段连接x,y对应元素定义的数据点。倘若它们连接所得折线不封闭,那么MATLAB将自动把该折线的首尾连接起来,构成封闭多边形。

c(或c1等)若取表5-1中色彩字符串,则该多边形用该字符所代表的颜色填

充。若取与 x,y 同维向量,则用c 向量元素在当前色板上所对应的色彩定义相应数据点(即多边形顶点)的颜色,而该多边形内部颜色由它们插补而成。

调用格式中的三个参变量(x,y,c)也可以是矩阵。

【例 5.7.1-1】 平面多边形填充(如图 5.7.1-1 所示)。

```
clf
x=rand(3);y=rand(3);    %x,y 矩阵每行为三角形的位置坐标
c=eye(3);               %c 为颜色矩阵,红绿蓝颜色插值
fill(x,y,c)
```

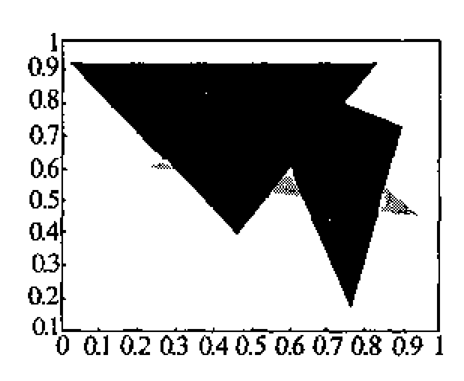


图 5.7.1-1　平面多边形填充

三维多边形填充

三维多边形的填色原理与二维情况相同,命令的调用格式也基本相同,这里举一个例子说明可通过 x-y 平面、进行对空间曲面片的操作。此方法不但可用于填充,还可处理曲面求交等计算机图形学中的复杂问题。

【例 5.7.1-2】 由 x-y 平面对三维空间中的多边形填色(如图 5.7.1-2 所示)。

```
clf
x=-4:4;y=x
[X,Y]=meshgrid(x,y);          %生成网格点的 x-y 坐标
Z=X.^2+Y.^2;
mesh(X,Y,Z);colormap([1 0 0]);hold on
x1=[2 ,2 ,3,3]';y1=[3 ,2 ,2 ,3]';
z1=x1.^2+y1.^2;               %由 x-y 平面坐标生成三维空间中多边形
fill3(x1,y1,z1,'m');
Z0=zeros(size(x));
plot3(X,Y,Z0,'ko')            %绘制网格点在 x-y 平面上的投影
stem3(X,Y,Z,'fill')
```

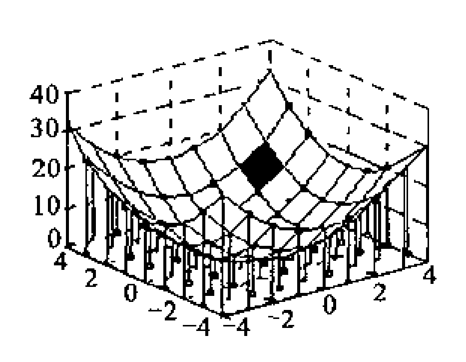


图5.7.1-2 空间多边形填充

5.7.2 色彩的控制与表现

颜色在图形的表现中起着非常重要的作用。它不仅给人以美的享受，而且从数学上讲，它给几何坐标系增加了一维的表现能力。在工程计算中，例如有限元计算，用颜色表现物体内部的应力，使问题更加直观、生动。本章将更深入地介绍由mesh，surf，pcolor 等命令所进行图形着色和渲染的能力。

色彩的调制

从计算机显示原理来讲，显示的颜色是由于显示器中的三色枪（红、绿、蓝）轰击屏幕时，荧光粉发出的光。那么，MATLAB 和其他图形软件一样，只用这三种基色（红R、绿G、蓝B）来绘图。三色枪的光束亮度比不同便绘出不同的颜色，这样颜色和数字就会有一一对应的关系。一种色彩用一个三元数组表示，该三元数组为[R G B]，元素R，G，B 在0和1之间取值，数值大小分别表示红、绿、蓝基色的相对亮度。通过对R，G，B 大小的不同设置，就可显示出不同的颜色。表5-4 给出了一些常用颜色所对应的三元数组的取值。

表5-4 **常用色三元数组**

基色三元数组			
红(R)	绿(G)	蓝(B)	颜色名称
1	1	1	白色(White)
0.5	0.5	0.5	灰色(Gray)
0	0	0	黑色(Black)
1	0	0	红色(Red)

续表

基色三元数组			
红(R)	绿(G)	蓝(B)	颜色名称
0	1	0	绿色(Green)
0	0	1	蓝色(Blue)
1	1	0	黄色(Yellow)
1	0	1	品红色(Magenta)
0	1	1	青色(Cyan)

这三元数组通过如下指令而变成当前绘图颜色：

colormap([R G B]) 设置当前绘图颜色。

色图

色图是一个储存在系统内部的颜色矩阵($m\times3$)，它的元素在[0,1]闭区间中取值，它的每一行代表一种颜色。色图既可以直接通过三元数组定义，也可以按某种数学规律生成。通过色图可绘出五颜六色的图形。

色图操作有以下方式：

colormap(Cm)	用 Cm 色图矩阵设置当前图形的色图。
colormap('default')	采用缺省色图 hsv。
map=colormap	获取当前图形所使用的色图矩阵。

【例 5.7.2-1】 用给定的色图矩阵设定色轴，绘制图形(如图 5.7.2-1 所示)。

```
clf
Cm=[1 0 0 ;0 1 0;0 0 1];          %指定红、蓝、绿三色色图矩阵
xx=[-2:0.125:2]';yy=[-1:0.125:3]';
[x,y]=meshgrid(xx',yy');Z=(x.*x)+(y.*y);
mesh(Z);                          %画 Z 阵的网线图，由 Z 阵的值决定着每个曲面片的
颜色
colormap(Cm);                     %根据 Cm 及 caxis 为当前图形生成色图
caxis([0,13]);                    %设置色轴刻度，将 0~13 分成三份进行着色
colorbar('horiz')                 %绘制水平色轴
```

色图函数

直接通过三元数组定义色图是比较麻烦的，MATLAB 对于常用的色图可由函数来生成，这样是很方便的(见表 5-5)。

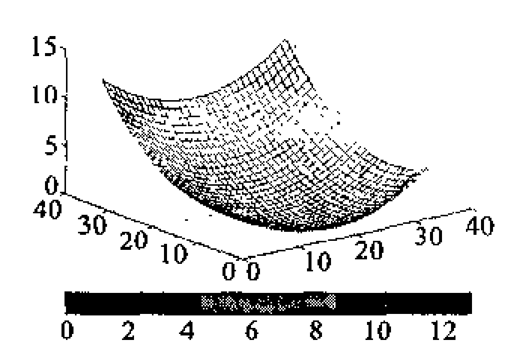


图 5.7.2-1　色图、色轴和着色的关系

表 5-5　　常用色图函数

色图函数名	典型颜色名称
cool	冷色
copper	纯铜色
gray	灰色
hot	暖色
hsv	Hsv 方式色图
pink	淡粉红色图
prism	光谱色图

假如对绘图的色图不作专门设置，那么默认的缺省色图是 hsv。它有下面两种使用格式：

hsv　　　　色图矩阵维数为(64 * 3)。

hsv(m)　　　色图矩阵维数为(m * 3)。

其他色图函数的使用方法与 hsv 相同。

【例 5.7.2-2】　产生 hsv 色图。

```
C=[1:7;1:7]';            %2×7 维着色矩阵
Cm=hsv(6)                %生成 6 级 hsv 色图矩阵，如下面显示
colormap(Cm);            %用 6 级 hsv 色图矩阵生成当前图形色图
subplot(1,3,1);          %第一幅子图
pcolor(C);caxis          %按矩阵 C 绘制伪彩图，缺省设置色轴
C=[1:16;1:16]';
Cm=hsv(16);              %生成 16 级 hsv 色图矩阵
colormap(Cm);            %用 16 级 hsv 色图矩阵生成当前图形色图
```

```
    subplot(1,3,2);          %第二幅子图
    pcolor(C);caxis([1,16]) %按矩阵C绘制伪彩图,设置色轴
    C=[1:32;1:32]';
    Cm=hsv(32);              %生成32级hsv色图矩阵
    colormap(Cm);            %用32级hsv色图矩阵生成当前图形色图
    subplot(1,3,3);          %第三幅子图
    pcolor(C);caxis([1,32]) %按矩阵C绘制伪彩图,设置色轴
Cm =
    1     0     0     %红
    1     1     0     %黄
    0     1     0     %绿
    0     1     1     %青
    0     0     1     %蓝
    1     0     1     %品红
```

可看出伪彩图的颜色是由色图 Cm、色轴范围[Cmin,Cmax]、着色阵 C 三者共同决定的。

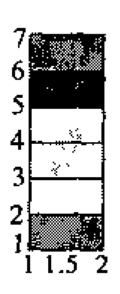


图 5.7.2-2 hsv 不同色图

5.8 曲面的表现

MATLAB 表现三维曲面有四种常用形式:线图(Mesh Plot)、渲染表面图(Shaded Surface Plot)、伪彩图(Pseudoccolor Plot)、等高线图(Contour)。其中,前两种方式还提供消隐、颜色、光线、视角等控制命令,从而使三维曲面的表现更加灵活自如。

5.8.1 重叠线的消隐

MATLAB 画 mesh 网线图时,对重叠曲面后面的网线采取了消隐措施,但有

时却需要用透视效果。为此,MATLAB提供了一个消隐开关。

hidden on　　消隐重叠线(缺省)。

hidden off　　透视重叠线。

5.8.2 曲面图着色

线框图通过着色网格线表现空间曲面的形状,而着色贴面图是根据色彩理论、光照理论对每一小曲面片着色来表现空间曲面的。MATLAB提供三个绘制着色表面图的指令:surf,surfc和surfl。其中surf是绘制着色表面图的基本命令,surfc绘制带等高线的着色表面图,而surfl可以控制光照效应。

surf指令的用法与mesh完全一样,它的调用格式是:

surf (Z,C)　以矩阵Z的下标为x-y坐标,按C指定的色彩绘制曲面图。

surf (X,Y,Z,C)　以矩阵X,Y元素为x-y坐标,按C指定的色彩绘制曲面图。

surf (x,y,Z,C)　分别以向量x,y元素为x-y坐标,按C指定的色彩绘制曲面图。

说明:

(1) 以上格式中,C是决定表面的每个张面色彩的矩阵(详见稍后关于shading命令的介绍)。当它缺省时,即默认C=Z。表面的每个张面按相应的Z阵元素值上色。

(2) 第二种调用格式中的X,Y可利用分格函数[X,Y]=meshgrid(x,y)从第三种调用格式中的x,y获得。

(3) 第三种调用格式中的x,y向量的长度必须分别等于Z阵的列维和行维。

【例5.8.2-1】 曲面图(如图5.8.2-1所示)。

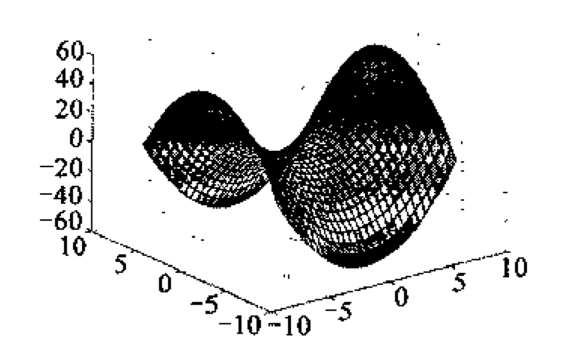


图5.8.2-1 曲面图

```
x=-7.5:0.5:7.5;y=x;
[X,Y]=meshgrid(x,y);
Z=X.^2-Y.^2;                    %绘制双曲抛物面
surf(X,Y,Z)
```

5.8.3 带光照效果的贴面图

surfl(x,y,Z,S,K)　　产生带光照的明暗处理三维表面图。

说明：

与 surf 指令一样，x、y、Z 定义三维表面，x、y 可以缺省。

S 确定光源位置，它可以由直角坐标定义：S=[sx,sy,sz]；也可以由球坐标中的方位角和俯视角定义：S=[az,el]，S 可以缺省。缺省时，光源的方位角在观察点逆时针方向 45 度处。

K 是确定用光模式的向量，K=[ka,kd,ks,spread]。在此，ka 是背景光(ambient)区域大小；kd 为漫射光(diffuse)区域大小；ks 为定向光(specular)区域大小；spread 是扩散系数。K 一般可缺省。

5.8.4 物体表面的渲染

函数 shading 设定由 mesh,surf,pcolor,fill 和 fill3 所创建图形时的渲染方式。函数的调用有三种方式：

shading flat　　图形渲染为平坦方式，将小曲面片按网格线段取同一对应的颜色。

shading interp　　插补方式，将小曲面片按网格线段双向进行颜色插值，线段的颜色是渐变的。

shading faceted　图形渲染为刻面，是缺省方式。它是在 shading flat 的基础上加上黑色网格线。

【例 5.8.4-1】 使用 interp 模式画图(如图 5.8.4-1)。

```
clf;x=-8:0.5:8;y=x;[X,Y]=meshgrid(x,y);
Z=sqrt(X.^2+Y.^2);
surf(Z);colormap(hsv);
shading interp ;
% light('Position',[0 0 1]);
```

5.8.5 等 高 线

等高线(Contour)是表现三维曲面的另一种形式。MATLAB 支持二维和三维等高线的图形绘制。在绘制时，用户可指定等高线的条数或位置。

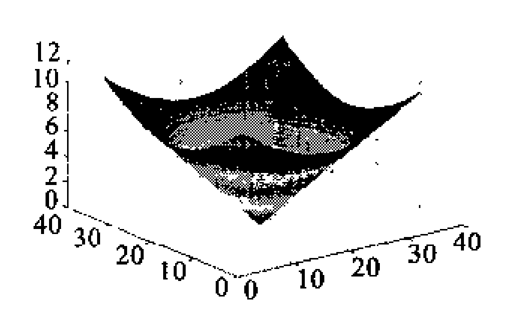


图 5. 8. 4-1　用 interp 模式绘图

在平面上绘制等高线

绘制平面等高线有关命令：

contour(x,y,Z,n)	绘制 n 条等高线。
contour(x,y,Z,v)	在向量 v 指定的值上绘制等高线。
C=contourc(x,y,Z,n)	计算 n 条等高线的 x-y 坐标数据。
C=contourc(x,y,Z,v)	计算向量 v 所指定的等高线的 x-y 坐标。
clabel(C)	给 C 阵所表示的等高线加注高度标识。
clabel(C,v)	给向量 v 所指定的等高线加注高度标识。
clabel(C,'manual')	对鼠标击中的等高线加注高度标识。

说明：

由 contourc 所给出的等高线坐标阵 C 是一个行数为 2 的矩阵。它的排列方式是每一条线由该矩阵的一个全行子阵表示：子阵的第一行第一列是等高线的高度值；子阵的第二行第一列是画该等高线的数据点数；子阵的其余列中的每一列都是一个数据点。

【例 5. 8. 5-1】 在二维平面上绘制 peaks 函数的 6 条等高线(如图 5. 8. 5-1 所示)。

```
clf
Z=peaks;
contour(Z,6)
c=contourc(Z,6);  %计算等高线坐标数据
clabel(c)         %给等高线加注标识
```

带等高线的贴面图

绘制带等高线贴面图的指令 surfc，用法与 meshc 一样，这里不再赘述。

【例 5. 8. 5-2】 绘制双曲抛物面带等高线曲面图(如图 5. 8. 5-2 所示)。

```
clf;x=-8:0.5:8;y=x;[X,Y]=meshgrid(x,y);
```

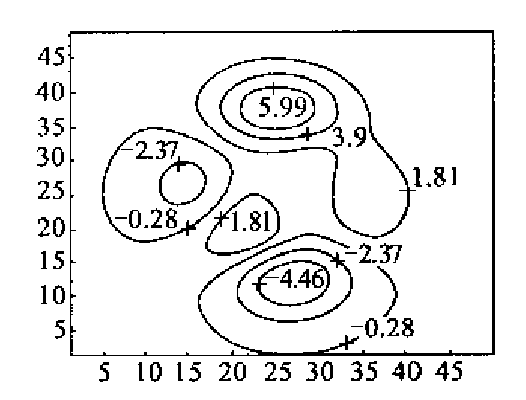


图 5.8.5-1 peaks 函数的 6 条等高线

```
Z=4*X.^2-2*Y.^2;
surfc(X,Y,Z)          %带等高线的着色表面
```

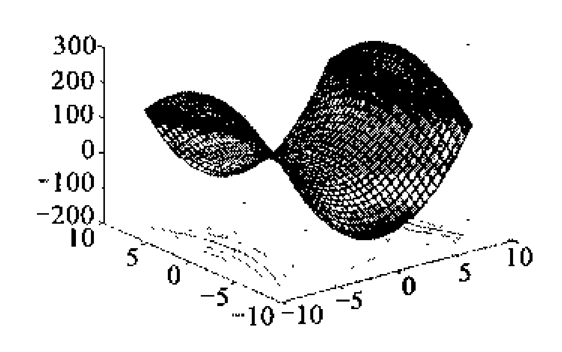


图 5.8.5-2 双曲抛物面及等高线

在三维空间中绘制等高线

在三维空间中绘制等高线的指令是 contour3，用法与 contour 类似，这里不再重复。

【例 5.8.5-3】 在三维空间中绘制双曲抛物面的等高线(如图 5.8.5-3)。

```
clf;x=-8:0.5:8;y=x;
[X,Y]=meshgrid(x,y);Z=X.^2-Y.^2;
contour3(Z,20);
```

5.8.6 二元函数的伪彩图

前面介绍的线框图和曲面图都是用三维坐标去表现二元函数 $z=f(x,y)$ 的。在这两种图形中，函数值表示三维空间实体中的标高。本节将介绍一种用色彩表现

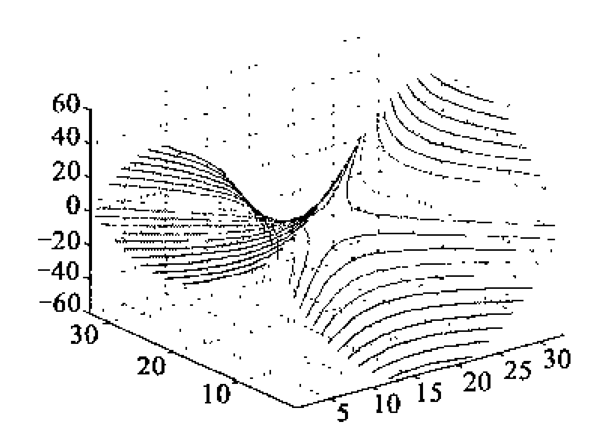


图 5.8.5-3　双曲抛物面的等高线

函数值的命令——伪彩图。伪彩图命令的调用格式如下：

pcolor(Z)　　　以Z阵元素的下标为x-y坐标画伪彩图。

pcolor(X,Y,Z)　以X,Y阵的元素为x-y坐标画伪彩图。

pcolor　　　　等价于surf (x,y,0*Z,Z)。

在此所说的"等价"是指，当观察角指令取view([0 90])时，所看到的"俯视"贴面图就是伪彩图。Z阵中的最大值和最小值分别对应色图(Color Map)的第一种和最后一种颜色。Z阵中的其他元素值分别对应色图的某种颜色，从而不同大小的元素便通过不同的颜色表现出来。

【例 5.8.6-1】　双曲抛物面的伪彩图(图 5.8.6-1)。

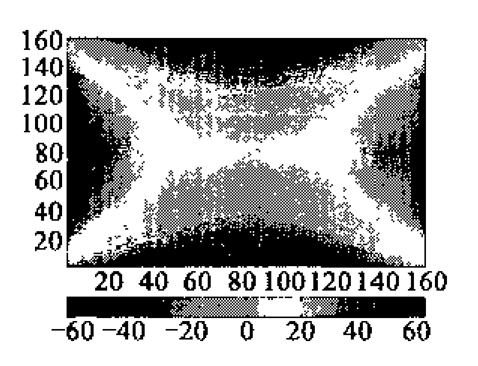


图 5.8.6-1　双曲抛物面的伪彩图

```
clf;x=-8:0.1:8;y=x;
[X,Y]=meshgrid(x,y);Z=X.^2-Y.^2;
pcolor(Z);
shading flat;              %采用flat着色模式,使网格线消隐
```

```
colormap;                %采用cool 色图
colorbar('horiz');       %画水平色轴，显示了数值与颜色的对应关系
```

5.8.7 矢量场图

矢量场图是由指令 quiver 实现的。它主要用于描写函数 $z=f(x,y)$ 的梯度大小和方向。一般调用格式为：

quiver(X,Y,DZX,DZY,s,'C')

(1)DZX,DZY 是该指令调用格式中必不可少的两个输入量，它们决定矢量场图中各矢量的大小和方向。它们可用二元函数矩阵的梯度命令 [DZD,DZY]=gradient(Z,dx,dy)求出。这里，dx,dy 是 x,y 方向上的计算步长。DZX,DZY 分别是对 x,y 偏导 。

(2)前两个输入量 X,Y 是 Z 阵元素的坐标矩阵，它们与"网格"向量的关系如下：

[X,Y]=meshgrid(x,y)

在矢量场图命令 quiver 中，X,Y 也可以用"网格"向量 x,y 替代。

(3)第五个输入量 s 是指定所画箭头大小的。缺省时，认为 s=1。

(4)第六个输入量'C'是字符串，确定线型和色彩。

【例 5.8.7-1】 等高线和矢量场图(图 5.8.7-1)。

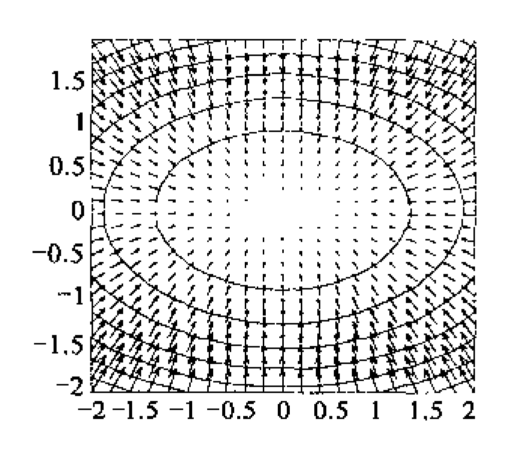


图 5.8.7-1 等高线和矢量场图

```
[x,y]=meshgrid(-2:.2:2,-2:.15:2);  %用于构造曲面的栅格坐标
z=sqrt(4-x.^2/9-y.^2/4);
[dx,dy] = gradient(z,.2,.2);  %dx,dy 分别是二元函数对 x,y 的偏导
contour(x,y,z), hold on                        %等高线
quiver(x,y,dx,dy,1.5), hold off, axis image    %矢量场图
```

说明：小箭头表示梯度，它是根据命令 gradient 求得偏导数画出的。

5.8.8 四维图形

对于定义 x-y-z 坐标的三元函数，若附加上色彩可实现四维表达，这样也可描述一个物体的属性如温度、内应力等。下面将介绍，定义在一般 x-y-z 坐标上的四维可视化命令。

为实现三元函数 $V=f(x,y,z)$ 的可视化表现，MATLAB 提供了一个绘制三维物体切片图命令及与之配合使用的三维网格坐标生成命令。

[X,Y,Z]=meshgrid(x,y,z)　　　　三维网格坐标的生成。

Slice(X,Y,Z,V,xi,yi,zi,n)　　　　绘制三维物体切片图。

其中：

x,y,z 决定"网线"位置，分别是(1 * n)，(1 * m)，(1 * p)向量。

X,Y,Z 三维网格坐标，它们都是(n * m) * p 维数组。

V 在网线结点上的三元函数值数组维数也为(n * m) * p。

xi,yi,zi 分别决定垂直于 x,y,z 轴切面位置向量。它们的维数可以各自不同。当取 0 维空阵"[ ]"时，表示没有切面存在。

切片上函数值的大小用色轴上对应的颜色表示。V 数组中的最大有限值和最小有限值定义了色轴的范围。由于切片位置可任意设置，因此 slice 通过三维坐标点上的色彩变化把图形的表现能力扩展到了四维。

【例 5.8.8-1】 函数 $v=4ye^{-(x^2+y^2+z^2)}$ 的四维图形(如图 5.8.8-1)。

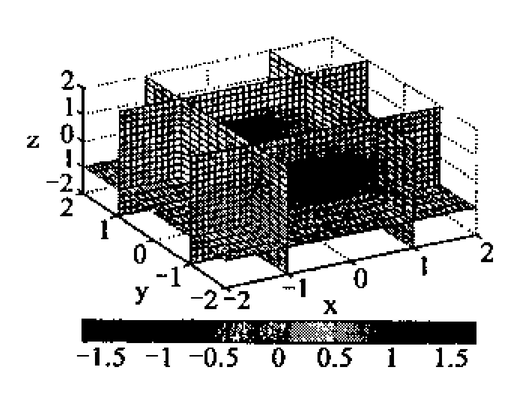


图 5.8.8-1　四维图形

```
x=-2:0.1:2;y=-2:0.25:2;z=-2:0.25:2;
[X,Y,Z]=meshgrid(x,y,z);
V= 4*Y.*exp(-X.^2-Y.^2-Z.^2);
xi=[-1,1];yi=[-1,1];zi=-1;        %选择切面位置
```

```
slice(X,Y,Z,V,xi,yi,zi);                %画切面图
xlabel('x');ylabel('y');zlabel('z');hold on
colorbar('horiz')                       %第四维坐标，即色轴
view([-30 45])                          %观察角控制
```

色轴在图的最下方，它标注着颜色和数值的对应关系。

5.9 交互式绘图

5.9.1 图形缩放

在科学分析和工程设计中，常常需要对计算所得的图形的"细部"进行观察，将图形放大或缩小。MATLAB 考虑到这种需求，设计了图形缩放命令。它的调用格式是：

zoom on　使图形窗处于缩放状态。

zoom off　在当前图形窗下恢复普通状态，即非缩放工作状态。

zoom out　使当前图形对象恢复到缩放前的大小。

zoom　在 zoom on 和 zoom off 两个状态间切换。

如果窗口在缩放状态下：

点击窗口

移动鼠标到图形窗下，单击鼠标左键一次，则图形将被放大一倍；若点击右键，那么图形将恢复到上一次放大前的大小。

拖拉矩形

把鼠标移到图形需要放大的附近，按住鼠标左键并使鼠标往右下方移动，会出现一个矩形框，使矩形框包容待放大部分，放开左键便显示框内图形。可在【例 5.9.2-1】中得以练习。

5.9.2 图形坐标点的输入

在 MATLAB 图形窗中可以用 ginput 命令来获取当前图形坐标点的数据，它的主要使用格式有：

[x,y]=ginput(n)　在当前图形窗中，可选定 n 个点，并返回两个 n 维坐标数据向量。

[x,y]=ginput　在当前图形窗中，可选任意多个点，并返回两个相应维数的坐标数据向量，当需要结束选点过程时，必须按 Enter 键。

【例 5.9.2-1】 由鼠标输入型质点后绘出插值曲线。

```
clf,disp('请用鼠标左键输入型质点,结束请回车')
[xi,yi]=ginput;
plot(xi,yi,'-*r');hold on
i=size(xi);t=1:i;
ts=1:0.1:i;
xs=spline(t,xi,ts); %样条插值
ys=spline(t,yi,ts);
plot(xs,ys,'b');
hold off
disp('按左键可放大图形,右键可恢复图形')
zoom on
```

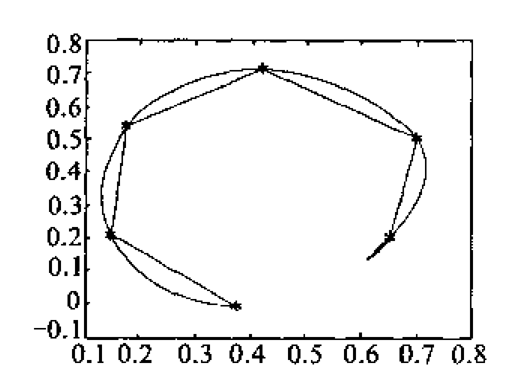


图 5.9.2-1 鼠标输入型质点后生成插值曲线

5.10 句柄图形

前面所论及的所有绘图命令都处于MATLAB图形系统中的高层界面。实际上MATLAB还提供了一些用于对线、面、文字、图像等基本图形对象进行操作的命令。这些命令可以对各图形对象进行更为细腻、自由的修饰和控制,不仅可以产生较复杂的图形,而且为动态图形的制作奠定基础。由于绘图时用到句柄,所以常称之为句柄图形。

5.10.1 句柄图形对象及关系结构

以往的命令一般是对整个图形进行操作,但在句柄图形中,所有的图形操作都是针对图形对象而言的。所谓图形对象是指图形系统中最基本、最底层的组元。具

体地说，图形对象是指：根屏幕(Root Screen)、用户窗口(Figures)、用户界面控制(User Interface Controls)、轴(Axes)、用户界面菜单(User Interface Menus)、线(Lines)、块(Patches)、面(Surfaces)、图像(Images)、字(Texts)。底层指令使用户有可能对图形的一个或几个对象进行独立的操作，而不影响图形的其他部分。正是这种功能为图形操作提供了极大的灵活性。

MATLAB 的图形对象体系可用图 5.10.1-1 表示。

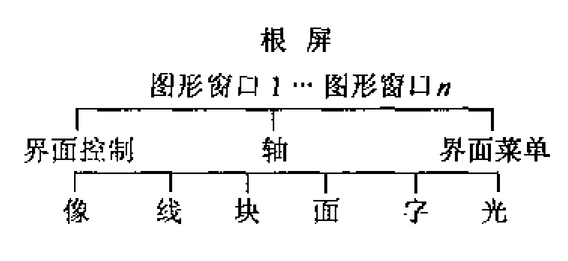


图 5.10.1-1 图形对象关系结构

根屏幕(Root)作为 windows 下的一个应用部分，是 MATLAB 整个树的根。在 MATLAB 图形系统中只有一个根，其他所有对象都是它的后代。

图形窗口(Figures)是根屏幕上窗口，窗口的数目不限。所有图形窗都是根屏幕的子代，除根屏幕外的其他对象则是窗口的后代。所有对象创建函数和高层作图指令都会建立一个图形窗口。当然，用户也可以用 figure 命令直接创建一个图形窗口。

用户界面控制(Uicontrols)即对 windows 下子窗的控制，包括对话框、按钮、滚动条等等的建立和操作。它是图形窗的子辈，与轴平行，独立于轴。

用户界面菜单(Uimenus)允许用户在图形窗口的上方建立用户菜单。它也是图形窗的子辈，与轴平行，独立于轴。

轴(Axes)是在窗口内建立一个子窗，并对此进行定位，其子辈对象的位置决定于轴所建立的坐标系。轴是窗的子辈，是线、面、块、像、字的父辈。所有对象创建函数和所有高层作图命令都能产生一个轴。轴也可以直接由 axes 命令创建。

线(Lines)是创建大多数二维图形和一部分三维图形的基本组元。线对象可以由函数 polt，polt3，contour 和 contour3 命令生成。

面片(Patches)是填色多边形，它也是轴的子辈。它可以用单色或插补色进行渲染。fill 和 fill3 建立块对象。

面(Surfaces)是矩阵数据的三维空间表现。它由许多四边形组成，四边形的顶点又由所给数据定位。面可以用单色或插补色表现，也可以仅用点间连线表现。它也是轴的子辈。pcolor，mesh，surf 类命令都能创建面。

像(Images)是矩阵元素直接映射到当前的色图上所得的结果。像有自己的色图。像是一个二维图形，它也是轴的子辈。像由函数 image 创建。

字(Texts)即为字符串。它们也是轴的子辈。

5.10.2 图形对象的创建

如果有在 windows 下的编程经验，就会知道，必须对每个对象赋予不能重复的识别标记(常称为句柄)。所有能创建图形对象的 MATLAB 函数都要给出所创对象的句柄，用 contour 命令产生的等位线中的每一条，都有自己的句柄，这样才能进行标注。一般用数字作为句柄，根屏幕的句柄总是零，图形窗口的句柄为某一整数，而其他对象的句柄是浮点数。需要注意的是：由于精度原因，最好把浮点数句柄赋给变量。

在 MATLAB 中，除根屏幕外，所有对象都由与之同名的内部函数创建。表5-6列出了这些函数的功能及调用方式，每一个函数都返回相应的句柄 h。

表 5-6 创建对象的主要函数

函数名	功能	调用格式举例
figure	创建图形对象	h=figure(n)
uicontrol	用户界面控制	h=uicontrol('property1',value1,…) property 和 value 指定界面的属性和属性值
uimenu	用户界面菜单	h=uimenu('property',value) property 和 value 为菜单选项及属性值
axes	创建轴对象	h=axes('position'[left,bottom,width,heighth]) 定义坐标轴的位置和尺寸
line	画线	h=line(x,y,z)绘制向量 x,y,z 确定的直线，如果不指定 z，则在 x-y 平面上画线
patch	填充多边形	h=patch(x,y,z,c) x,y,z 定义多边形，c 指定填充颜色
surface	创建三维曲面	h=surface(x,y,z,c) x,y,z 定义三维曲面，c 是色彩矩阵
image	显示图像	h=image(x) x 为图像矩阵
text	标注文字	h=text(x,y,'string') x,y 指定字符串 string 的标注位置

每个函数只能创建一个图形对象，并将它们置于适当的父辈对象之中。例如，line 命令将在当前轴上用缺省数据画“线”。假若此指令运作前，“轴”、“窗”不存在，则 MATLAB 会自动创建它们。假若“轴”、“窗”已经存在，那么这“线”将被画在已有的“轴上”，且不影响该轴上已有的其他对象，这与前面讲过的高层命令不同。

【例 5.10.2-1】 生成一个面，并比较与 5.5.2 节讲过的 surf 命令有何不同（图 5.10.2-1 所示）。

```
[x,y]=meshgrid([-2:.4:2]);
Z=(x.^2-y.^2);
%生成图形窗口句柄
fh=figure('Position',[100 255 500 400],'Color','W');
%生成轴句柄
ah=axes('Color',[.7 .7 .7],'XTick',[-2 -1 0 1 2],...
'YTick',[-2 -1 0 1 2]);
sh=surface('XData',x,'YData',y,'ZData',Z,...    %生成面
'FaceColor',get(ah,'Color')+0.2,...    %将轴的颜色变浅
'EdgeColor',[.5 .5 .5],'Marker','*',...
'MarkerFaceColor',[0 1 0]);
view(3)
```

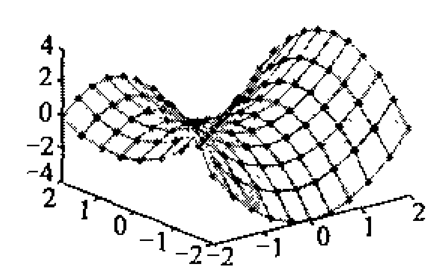


图 5.10.2-1　用句柄生成的图形

对象句柄的操作

在【例 5.10.2-1】中已经看到：通过命令的调用格式 sh＝surface，在对象创建的同时，获得创建对象 surface 的句柄值 sh。这是获得对象句柄的一种方法。MATLAB 还提供两个专门用于获取对象句柄的函数：

gcf　　返回当前图形窗口的句柄。

gca　　返回当前轴的句柄。

gco　　返回当前对象的句柄。

delete　　删除句柄对应的对象。

它们既可以赋予某一个变量以供后用，也可直接作为其他函数的输入变量使用。

【例 5. 10. 2-2】　改变【例 5. 10. 2-1】的曲面位置、颜色、型质点等属性（图 5. 10. 2-2）。

```
set(sh,'XData',x+5,'YData',y+5,'ZData',Z,...   %改变位置
'FaceColor',get(ah,'Color')-0.3,...      %将轴的颜色加深
'EdgeColor',[.1 .1 .1],'Marker','+',...   %改变点的标记
'MarkerFaceColor',[1 0 0]);
pause
delete(gca)                   %删除轴以下的子实体
```

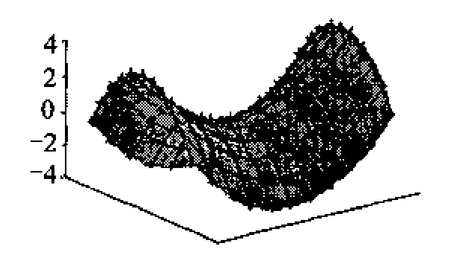


图 5. 10. 2-2　编辑曲面属性

5. 10. 3　对象属性的设置和获取

所有对象都有决定其如何表现的属性。对象属性可分为两类：一类是“共性”，包括它的父辈和子代、类型、是否可视、剪辑、中断允许等；另一类是“特性”，如“轴”的刻度、定义“面”的数据等。

当对象被创建时，该对象的属性就被一组缺省值初始化。

用户既可以(用 get 指令)获取任何属性值，也可以(用 set 指令)设置大多数的属性值，且设置仅适用于本对象，而对其他同类对象的属性值没有影响。

MATLAB 提供两种设置对象属性的方法：一种是在对象创建时设置其属性；另一种是通过函数 set 重新设置已有对象的属性。

获取函数 get

get(h)　　　　　　　　获取 h 对象所有属性的当前值。

get(h,'PropertyName')　　获取 h 对象 PropertyName 所指定属性的当前值。

【例 5. 10. 3-1】　获取【例 5. 10. 2-1】所创建“面”的属性。

```
get(sh,'FaceColor')
```

```
ans =
0.9000    0.9000    0.9000
```

读者也可试一试命令 get(sh)，并观看结果。

设置函数 set

set(h)	显示 h 对象所有可设置的属性及取值。
set(h,'PropertyName')	显示 h 对象(PropertyName)指定属性的可取值。
set(h,'PropertyName',Property Value,…)	设置 h 对象(PropertyName)指定属性的值。

【例 5.10.3-2】 通过重新设置【例 5.10.2-1】的"面"属性面获得的瀑布线图（图 5.10.3-2 所示）。

```
set (sh,'FaceColor','w','MeshStyle','row')
```

注意：如果要编辑 sh 句柄图形，要求系统中必须保留 sh 句柄图形。

如果要在同一窗口生成多个图形，不必用 subplot 等分窗口，而是用句柄根据需要任意划分窗口。

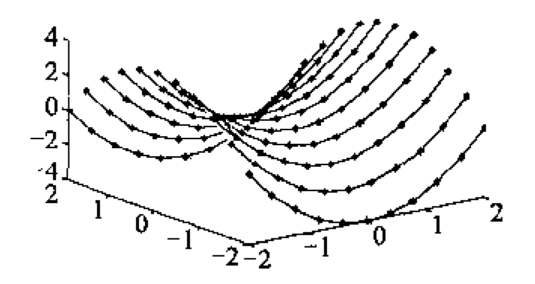


图 5.10.3-2 通过重新设置"面"属性而获得的瀑布线图

【例 5.10.3-3】 在同一窗口多个轴分别生成图形。

```
[x,y]=meshgrid(-8:.5:8);
z1=x.*x-y.*y;
z2=x.*x+y.*y;
z3=sqrt(x.*x+y.*y);
z4=x+y;
[x5,y5,z5]=sphere(30);
h(1)=axes('Position',[0 0 1 1]);
mesh(x,y,z1);
h(2)=axes('Position',[0 0 0.4 0.4]);
mesh(x,y,z2);
h(3)=axes('Position',[0.6 0.6 0.4 0.4]);
mesh(x,y,z3);
```

```
h(4)=axes('Position',[0.6 0 0.4 0.4]);
mesh(x,y,z4);
h(5)=axes('Position',[0 0.6 0.5 0.5]);
mesh(x5,y5,z5);
set(h,'Visible','off')
set(gcf,'renderer','painters')
```

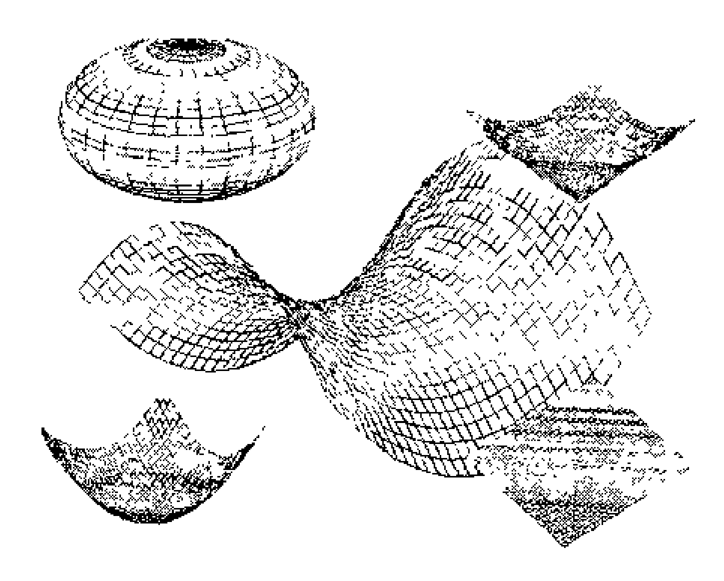


图 5.10.3-3 一窗多轴绘制图形

5.11 动画绘图

5.11.1 实时动画的制作

实时动画的应用很广，它可以进行物理仿真、数字仿真等，将真实情况模拟显示出来，以节省试验经费和时间，使设计、修改和试验同时进行，这是非常有意义的。

动画制作时要受到软硬件资源的限制。从硬件上讲，图形需占较多的计算机资源。大量的、高质量的图形操作要求计算机有很高的运行速度，较大的内存和显存，因此一个高性能的计算机在动画制作中是必不可少的。从软件方面讲，动画制作通常有两种方法：(1)预先将图形制作好，并放到图形缓冲区内，然后一帧一帧地播放。这种方法计算量大、占用内存多，要达到人眼连贯程度，播放速度至少 24 帧/s，这对计算机硬件是不低的要求。目前的多媒体技术、物理仿真及动画片制作就用这

种方法。(2)保持整个背景图案不变,只更新运动部分的图案,可以减少内存的需求量,加快图像的实时生成速度,这称为实时动画。很多需要编程的数字仿真都采用这种技术。这正是本节要介绍的内容。

改变某几个图形对象而不破坏其他部分图形的具体办法是:(1)计算活动对象的新位置,并在新位置上将它显示出来;(2)擦除原位置上原有对象,刷新屏幕;(3)重复步骤1和步骤2。

如何显示新对象,擦除旧对象,而又不破坏背景图案?这对一般计算机编程语言来说不是件轻松事。然而,这在MATLAB图形系统中却轻而易举,只需在创建图形对象时指定它的擦除方式(Erase Mode)便可。

MATLAB为Erase Mode属性设置了四个选项:

(1)normal方式 重画整个显示区。该模式产生的图形准确、但较慢。

(2)background方式 将旧对象的颜色变为背景颜色,会损坏被擦对象下面的对象。

(3)xor异或方式 对象的绘制和擦除由该对象颜色与屏幕颜色的异或而定。只画与屏幕色不同的对象点,擦除与屏幕色不同的原对象点。该方式不损害被擦对象下面的其他图像。

(4)none方式 不做任何擦除。

当新对象属性设置后,应刷新屏幕,使新对象显示出来。MATLAB刷新屏幕的命令是drawnow。drawnow命令迫使MATLAB暂停目前的任务序列而去刷新屏幕;若没有drawnow指令,MATLAB要等任务序列执行完后才会去刷新屏幕。

下面是两个实时动画制作实例。

【例5.11.1-1】 如图5.11.1-1所示半径为R车轮上质点作摆线运动,方程为

$$\begin{cases} x=V_0t-R\sin\dfrac{V_0}{R}t \\ y=R-R\cos\dfrac{V_0}{R}t \end{cases}$$

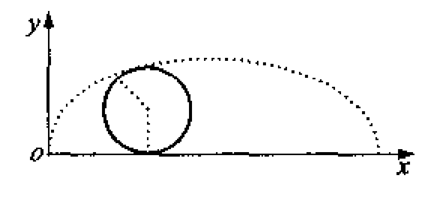


图5.11.1-1 摆线轨迹

编写程序演示其动态轨迹。

```
v0=2;R=1;t=0:0.009:10;
x0=0;y0=0;
x=v0*t-R*sin(v0/R*t);
y=R-R*cos(v0/R*t);
plot(x,y);  %绘制摆线
axis([0,20,0,10]);
axis('off');           %不显示坐标轴
point=line(x0,y0,'color','r','linestyle'          ...
'erasemode','xor','markersize',20);     % 生成质点句柄
%利用"线"对象创建小球、定义"线"色、"线"型、点的大小(20)、擦除方式(xor)
%小球对象的句柄值保留，赋给变量point
n=length(t);
i=1;
while 1
set(point,'xdata',x(i),'ydata',y(i));  %点的位置
drawnow;                               %刷新屏幕
  i=i+1;
if i>n,
  i=1;
end
end
```

【例5.11.1-2】 一动点沿一半径为a的圆柱，以u作匀速直线运动，同时以角速度ω作旋转运动(如图5.11.1-2)。动点运动方程为：$x=a\cos(\frac{2\pi}{T}t)$，$y=a\sin(\frac{2\pi}{T}t)$，$z=ut$。编写程序演示其动态轨迹。

```
T=1;
a=5;u=0.05;x0=0,y0=0,z0=0;
  [x,y,z]=cylinder(5,90);
  mesh(x,y,z);colormap([0 0 1]);
hidden off;
axis([-5,5,-5,5,0,1]);
axis('off');           %不显示坐标轴
head=line(x0,y0,z0,'color','r','linestyle','.',...
     'erasemode','xor','markersize',20);
```

```
t=0;
dt=0.005;
while 1
   t=t+dt;
   X=a*cos(2*pi/T*t);
   Y=a*sin(2*pi/T*t);
   Z=u*t;
set(head,'xdata',X,'ydata',Y,'zdata',Z);
drawnow;
if Z>1,
t=0;                    %返回循环原始点
end
end
```

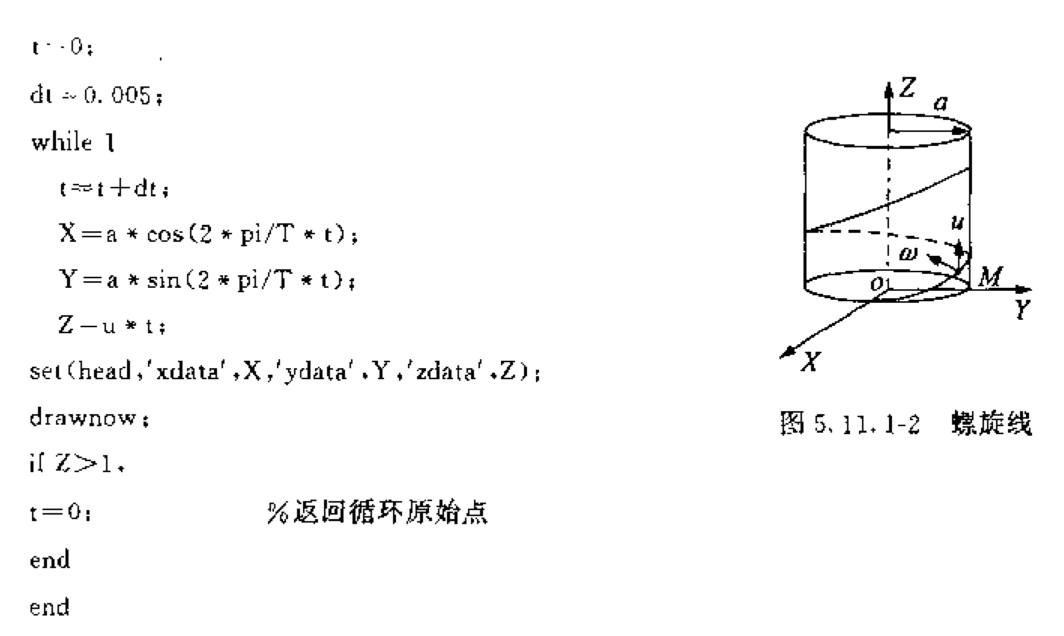


图 5.11.1-2　螺旋线

如果上面程序可以在 MATLAB 中运行，并在图形窗中看到动点作无休止地运动，可用 Ctrl+Break 中断程序。

习　题　五

1. 绘制下列各种函数图形。

(1)绘制下列极坐标图形(区间 $0\leqslant\theta\leqslant 2\pi$)：

① $r=2(1-\cos\theta)$

② $r=3(1+\cos\theta)$

③ $r=3(1+\sin\theta)$

④ $r=\cos 3\theta$

(2)求函数 z 的三维图形。定义区间与 z 函数表达式如下：

$$z=\frac{1}{(x+1)^2+(y+1)^2+1}-\frac{1.5}{(x-1)^2+(y-1)^2+1};\ 5\leqslant x\leqslant 5,-5\leqslant y\leqslant 5.$$

2. 用 MATLAB 命令绘制下列数学函数的图形。

(1)绘制如下函数图形：

$y(t)=1-2e^{-t}\sin(t)\quad(0\leqslant t\leqslant 8)$

且在 x 轴上写"Time"标号，y 轴上写"Amplitude"标号，图形的标题为"Decaying Oscillating Exponential"。

(2)绘制如下图形：

$y(t)=5e^{-0.2t}\cos(0.9t-30°)+0.8e^{-2t}\qquad(0\leqslant t\leqslant 30)$

(3)在 $0\leqslant t\leqslant 10$ 区间内绘制下面图形：

$y(t)=1.23\cos(2.83t+240^\circ)+0.625$

$x(t)=0.625$

在同一个图中绘制上述函数，求出 $y(t=0)$ 和 $y(t=10)$ 的点。注意弧度与角度的区别。

(4)在 $0\leqslant t\leqslant 20$ 区间内，且在同一图中绘制下述函数图形：

$y_1(t)=2.62e^{-0.25t}\cos(2.22t+174^\circ)+0.6$

$y_2(t)=2.62e^{-0.25t}+0.6$

$y_3(t)=0.6$

在 y 值对应的 -2 到 3 的区域内，首先求 y_1 的最小值与最大值，然后再求次大值与次小值。

(5)对应 $0\leqslant t\leqslant 25$ 区域内，在同一图中绘制下列函数：

$y_1(t)=1.25e^{-t}$

$y_2(t)=2.02e^{-0.2t}$

$y_3=2.02e^{-0.2t}\cos(0.554t-128^\circ)+1.25e^{-t}$

限制在如下区域内：

$0.2\leqslant y\leqslant 1$　　　　$0\leqslant x\leqslant 16$

求 $y_3(t)$ 值：$y(t=0)$，y_{max}，y_{min} 和 $y(t=12)$。

3. 完成下列几何变换，熟悉不同变换矩阵的作用。

(1) $\boldsymbol{A}_1=\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$　　$\boldsymbol{x}=\begin{bmatrix}1 & 1\\3 & 1\\2.5 & 2\end{bmatrix}$

(2) $\boldsymbol{A}_2=\begin{bmatrix}-1 & 0\\0 & 1\end{bmatrix}$

(3) $\boldsymbol{A}_3=\begin{bmatrix}2 & 0\\1 & 2\end{bmatrix}$

(4) $\boldsymbol{A}_4=\begin{bmatrix}1 & 2\\1 & 0\end{bmatrix}$

(5) $\boldsymbol{A}_5=\begin{bmatrix}\cos(\pi/2) & -\sin(\pi/2)\\\sin(\pi/2) & \cos(\pi/2)\end{bmatrix}$

4. 用MATLAB命令构造一个有101个元素的矢量，它的元素交替为1和-1。使用plot和stairs命令分别绘制该矢量的图形。

5. 绘制一段圆弧，圆心角从0°到60°。

6. 一般二维曲面方程为 $z=\pm c\sqrt{d-\frac{x^2}{a^2}-\frac{y^2}{b^2}}$，画出下面不同情况的二次曲面：

(1) $a=5, b=4, c=3, d=1$

(2) $a=5\mathrm{i}, b=4, c=3, d=1$

(3) $a=5\mathrm{i}, b=4\mathrm{i}, c=3, d=1$

7. 求直线与曲线相交，曲线为 $3\sin x$，直线分别由下面条件决定：

(1) 方程 $y=2$；

(2) 两点(0,1)和(2,2)。

8. 分别生成6级、12级、24级prism色图，并观察彩图的颜色与色图Cm、色轴范围[C_{min},C_{max}]、着色阵C三者之间的关系。

第六章 MATLAB语言的程序设计

在第五章绘图时，我们对MATLAB语言有了一定的了解。本章将着重介绍MATLAB语言的使用方法。MATLAB语言是一种解释性语言、ASCII形式，可读性强，效率很高，比常用的BASIC，C，FORTRAN和PASCAL等语言好得多，被MathWork公司称为第四代编程语言。"一盎司MATLAB语言价值等于一磅的C和FORTRAN语言"。常常一两句MATLAB语言，有时需要几十行甚至上百行C或FORTRAN语言，所以MATALB在数值计算和图形显示方面有广泛的应用。

6.1 编 程 入 门

6.1.1 M文件的形式

MATLAB有两种常用工作方式：一种是命令行操作方式，另一种是M文件的工作方式。

M文件有两种形式：命令文件(Script File)和函数文件(Function File)。

命令文件

【例6.1.1-1】 计算$\sum_{n=1}^{20} n!$。

(1)用编辑器或记事本Notepad编写以下内容。计算1至20阶乘的和。

```
s=0;t=1;n=1;
while n<=20
t=t*n;
s=s+t;
n=n+1;
end
s
```

(2)选择[文件]下拉菜单中的[保存]选项，将所写文件存放于磁盘中，并起名为prodsum.m。

MATLAB指令窗中键入文件名prodsum，运行结束，可在屏幕上看到计算内容。

```
s =
  2.5613e+018
```

说明：

①符号“%”引导注释语句，MATLAB 不予执行。另外通过 help prodsum 命令可将此行显示在 MATLAB 命令窗内，供用户查询使用。

②MATLAB 程序不需要用“end”为 M 文件的结束标志。

③若用户把文件 prodsum.m 存放在自己的文件夹内或工作目录上（假如名为 d:\stu），那么在运行 prodsum.m 之前，应该先使“d:\stu”处于 MATLAB 的搜索路径上。方法是：在 MATLAB 指令窗中先键入“cd d:\stu”或“path”(path,'d:\stu')。

函数文件

【例 6.1.1-2】把【例 6.1.1-1】中的命令文件改成函数文件，并运行之。

(1)在编辑器中编写以下内容。

```
function f=prods(i)
%prods.m 计算 1 至 20 阶乘和的函数文件
%f=prods(i)
s=0;t=1;n=1;
while n<=i
t=t*n;
s=s+t;
n=n+1;
end
```

(2)将文件 prods.m 存盘。该文件定义了名为 prods 的新函数。此函数就成了 MATLAB 的宏命令，其用法与 MATLAB 其他函数一样使用。

(3)在 MATLAB 命令窗运行以下命令：

```
prods(20)
```

这样 20 传给了程序中的 i，结果与【例 6.1.1-1】相同。

第一行 function 的作用：指明该文件是函数文件；定义函数名 prods、输入参数 i 和输出参数 f。函数名可以是 MATLAB 中任何合法的字符。输入和输出的参数类型可以是数值，也可以是字符串，参数根据实际需要设置。

函数调用

在 MATLAB 中，函数文件可相互调用，甚至可调用它自己（即递归调用）。调用函数的形式是：

[输出参数 1，输出参数 2，…]= 函数名(输入参数 1，输入参数 2，…)

【例 6.1.1-3】　计算半径为 1～10 不同圆的面积和周长。

(1)建立函数文件area.m。

```
function [f1,f2]=area(r)
%计算圆面积和周长
f1=pi*r^2;
f2=2*pi*r;
```

(2) 编一个程序调用上述函数文件test.m。

```
R=10;           %输入圆半径
f1=zeros(1:R);f2=zeros(1:R);
for i=1:R
  [S(i),L(i)]=area(i);
end
S
L
```

在本例中,命令文件test对函数area调用一次,就传入参数r一次,便得到结果S和L。

递归调用

递归调用可使程序设计方便、简洁、凝练,在程序设计中占有重要地位。

【例6.1.1-4】 用递归调用形式计算二阶斐比拉契数列。

$$fib(i)=\begin{cases}1 & i=1,2\\ fib(i\text{-}1)+fib(i\text{-}2) & i>2\end{cases}$$

(1)编写递归调用函数文件fib.m。

```
function f=fib(i)
% 计算二阶斐比拉契数列 i>0
if (i==1)|(i==2)
f=1;
return;
else
f=fib(i-1)+fib(i-2);
return;
end
```

(2)运行程序函数文件。

```
  fib(9)
ans=
    34
```

参数传递

在函数文件中,可检查传入或传出参数数目,这样使编程具有很大的灵活性。

共函数如下：

nargin　　　检查在调用该函数文件时传入参数数目。

nargout　　　检查在调用该函数文件时传出参数数目。

【例 6.1.1-5】 以函数 plot.m 的意图说明 nargin 的用法(该函数不带输出参数,但有三个可选的输入参数)。

```
function test(x,y,x1,y1)
if nargin==0,
disp(' ??? Error using ==> plot')
disp(' Not enough input arguments.')
elseif nargin==1      %有一个输入参数即 nargin=1
plot(x);
elseif nargin==2      %有两个输入参数即 nargin=2
plot(x,y);
elseif nargin==3      %有两个输入参数即 nargin=3
plot(x,y,x1);
elseif nargin==4      %有两个输入参数即 nargin=4
plot(x,y,x1,y1);
end
```

6.1.2 数据类型

在 MATLAB 中共有六种基本的数据类型：双精度型(double)、字符型(char)、稀疏型(sparse)、8 位型(unit8)、细胞型(cell)和结构型(struct)。这六种数据类型的关系可如图 6.1.2-1 所示。

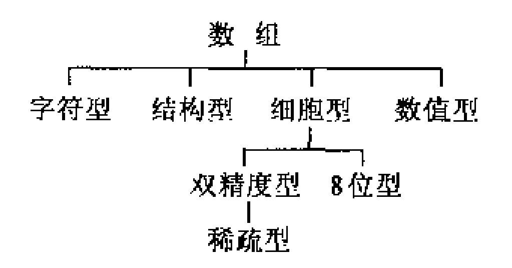


图 6.1.2-1 数据类型树

其中最常用的是双精度型和字符型。数组位于最上层，说明 MATLAB 的所有数据类型都是向量。表 6-1 为数据类型的举例。

表 6-1 数据类型举例

数据类型	举　例	说　明
双精度型(double)	[1 2;3 4];5+6i	用于MATLAB中数值、符号等大部分计算
字符型(char)	'56','Hello'	字符操作、字符运算
稀疏型(sparse)	speye(3)	稀疏矩阵
8位型(unit8)	unit8(magic(3))	图形处理
细胞型(cell)	{16 'stu' rand(3)}	数组中可用不同类型维数，用于编写大型软件
结构型(struct)	a.day=16;a.name='Li';	与C语言相似，用于编写大型软件

在MATLAB中，用户还可以自己定义数据类型，自定义类型和MATLAB内部的数据类型一样使用。

变量的数据类型可用函数isa来查看，其调用格式为：

class(OBJ)　　OBJ是变量名，查询OBJ数据类型。

isa (OBJ,'class-name')　　OBJ是变量名，'class-name' 是数据类型，例如'char','double','cell'。

isstr(str)　　验证是否字符类型。

【例6.1.2-1】 字符变量的输入和检查。

```
    a=' 3';
    class(a)
    isa(a,'char')
    isstr(a)
ans =
    char
ans =
    1
ans =
    1
```

命令isa,isstr用来检查变量是否是字符变量。检查变量是字符则返回1,否则返回0。

【例 6.1.2-2】　数值型变量的输入和检查。

```
    a=sin(pi/2);
    isa(a,'numeric')
    isa(a,'double')
    isa(a,'char')
    isstr(a)
ans =
    1
ans =
    1
ans =
    0
ans =
    0
```

6.1.3　全局变量及局部变量

局部变量常常是在函数文件的内部，它的作用域是这个函数文件，运行完毕后就消失。全局变量的作用域是整个 MATLAB 工作空间，直到 MATLAB 退出或用 clear 删除为止。全局变量可用于函数文件间的数据传递，条件是在整个运行过程中数据不能随意改变，否则就要出错。这样，它也会为结构化程序设计带来麻烦。

全局变量可用命令 global 定义，例如：

global　a　b　　a,b 就成为全局变量。

【例 6.1.3-1】　利用全局变量，建一个计算【例 6.1.1-2】1～20 阶乘和的无参数传递的函数文件，并用它计算。

(1)编写函数文件 prods.m。

```
function f=prods
global num
%prods.m 利用全局变量计算 1 至 20 阶乘和的函数文件
%f=prods(i)
s=0;t=1;n=1;
while n<=num
t=t*n;
s=s+t;
n=n+1;
end
```

(2)在 MATLAB 命令窗运行以下命令。

```
global num
num=20;
prods
```

这样20通过num传给函数文件,结果与【例6.1.1-1】相同。

```
s =
2.5613e+018
```

6.1.4 键盘输入和输出

与BASIC相似,MATLAB也可提示用户用键盘进行输入,并接受该输入。键盘输入命令input,yinput常用的格式如下:

(一) a=input('Please input a number:')

该命令运行后,将给出如下文字提示,并等待键盘输入:Please input number:

用户可以输入数字或表达式,也可以输入字符串(两端必须有单引号),按[Enter]键确认后,该输入被赋给变量a。

(二) a=yesinput('Prompt:',Defaut,limits)

yesinput这个输入命令可带有缺省输入和输入的范围检查。它的用法较多,在此举例介绍两种。

【例6.1.4-1】 带输入数值范围检查的yesinput用法。

第一步:在MATLAB命令窗中键入

a=yesinput('请输入数据' ,10,[6,15]);

第二步:该命令运行后,屏幕上有如下提示:

请输入数据(10):

第三步:等待用户输入数据。如果只是按回车键,则默认输入值为10;如果输入值大于15或小于6时,则认为输入无效,等待用户重新输入。

【例6.1.4-2】 带输入选项检查的yesinput用法。

(1) 在MATLAB命令窗中键入:

color=yesinput('请选择图中的颜色','red','red|blue|green');

(2) 该命令运行后,屏幕上有如下提示:

请选择图中的颜色(red):

(3) 等待用户输入数据。如果只是按回车键,则默认输入字符串为red;

如果输入字符串与选项内容不符,则认为输入无效,等待用户重新输入。

(三)[S,ERRMSG] = sprintf(FORMAT,A1,A2,...)

与C语言相似,例如:

```
sprintf('S=%6.3f',(sqrt(2)))
```

```
sprintf('%15.5f',1/eps)
sprintf('%0.5g',1/eps)
sprintf('%d',round(pi))
sprintf('%s','hello')
```

结果为：

```
S= 1.414
4503599627370496.00000
4.5036e+15
3
hello
```

6.1.5 程序流控制

echo 命令

echo 命令可使文件在执行时可见。

对命令文件其格式为：

echo on	切换到显示其后所有被执行命令文件指令的状态。
echo off	切换到其后所有被执行命令文件指令不被显示的状态。

下面的 echo 调用格式对函数文件、命令文件都适用。

echo FileName on	使 FileName 指定文件的指令在执行中被显示出来。
echo FileName off	终止显示 FileName 文件的执行过程。
echo on all	显示其后所有被执行文件的过程。
echo off all	使其后所有被执行文件的过程不被显示。

pause 命令

pause 命令使程序运行暂停，等待用户按任意键继续。pause 命令在程序调试以及需要看中间结果时特别有用。pause 的用法有两种：pause 暂停执行程序；pause(n)在继续执行前暂停 n 秒。

break 命令

break 语句一般是包含在 while，for 结构中，当条件满足时，可以使 while，for 结构的循环终止。如果没有 break，循环将无限制进行下去。

【例 6.1.5-1】 计算 1 至 10 的和。

```
echo off          %不显示运行过程
n=1
while 1           %无限循环
  if n>=10;
    break;
```

```
    end;
  n=n-1;
  end
  n
```

6.1.6 程序的优化

对于大型计算往往要花很长的时间,如果能优化内存管理,则会大大提高效率。方法是:

(1) 循环向量化

由于MATLAB对矩阵的单个元素循环时速度很慢,如果把循环向量化,不但能缩短程序的长度,而且能提高程序的执行效率。所以编程时,应尽量对矩阵或向量编程,而不是对矩阵的元素编程。

【例6.1.6-1】 循环向量化速度的比较。

```
format long
echo off
tic             %开启计时器计时
for k=1:10000
  Y(k)=exp(k);
end
toc             %关闭计时器显示时间
%循环矢量化的计时
tic
x=1:10000;
y=exp(x);
toc
elapsed-time =
  16.75000000000000
elapsed-time =              %后一种方法速度要快一些
  0.05000000000000
```

(2) 预定义变量

由于MATLAB文件是面向矩阵的语言,MATLAB将任何一个变量都看成一个矩阵。如果变量是一个数,就认为是1×1的矩阵。一般,变量不需要定义,但是预定义一个变量,可使程序在执行循环结构时的速度加快。

【例6.1.6-2】 预定义变量速度的比较。

```
format long
y1=zeros(1,10000);
```

```
    y2=zeros(1,10000);
    tic; for i=1:10000,y0(i)=exp(i);end;toc        %最慢
    tic; for i=1:10000,y1(i)=exp(i);end;toc        %较快
    tic; i=1:10000;y2=exp(i);toc                   %最快
elapsed-time =
  16.42000000000000
elapsed-time =
  0.38000000000000
elapsed-time =
    0
```

由于变量 y0 没有预定义，所以程序解释器在每次循环中都重新定义 y0 的维数。而 y1 和 y2 的预定义使程序做循环时，省略了定义维数这一步，速度要快些。而 y2 进行了循环向量化，速度最快。看来我们编程时，既要预定义维数，又要循环向量化。

至于如何进一步优化程序、提高效率，在 6.6 节及 6.7 节还有详细的介绍。

6.2 控制语句

MATLAB 语言和其他计算机语言一样也有程序结构：(1)顺序结构；(2)循环结构；(3)分支结构。

6.2.1 循环结构

MATLAB 语言提供了两种循环方式：for-end 循环和 while-end 循环。

for-end 循环

与 BASIC，C，FORTRAN，PASCAL 等其他计算机语言一样，循环条件是有规律地变化的，通常是把循环条件的初值表达式放在循环的开头，这种形式就是 for 循环结构。

for-end 循环的一般形式是：

```
for i =表达式
    语句体
end
```

通常，表达式是一个向量，如 m:s:n，s 是步长。该向量的元素被逐一赋给循环变量 i，然后执行语句体。

【例 6.2.1-1】 简单的循环。

```
for i=1:5
```

```
      x(i)= sin(i);
  end
  x
x =
0.8415    0.9093    0.1411    -0.7568    -0.9589
```

说明：

在1:5这个MATLAB的冒号结构中，循环初值是1，终值是5，缺省步长为1。循环体实现对x的5个元素赋值，环可以嵌套。需要注意的是，第一个for必须与end相匹配，否则程序将出错。

while-end 循环

While 循环使语句体在逻辑条件控制下重复若干次，直到循环条件为假。While 循环的一般形式是：

```
While 表达式
    语句体
end
```

只要表达式条件满足，语句体就重复执行。当表达式结果不是标量时，可以用 any，all 等函数处理。如果表达式为1，该循环将无限制地进行下去。

【例6.2.1-2】 利用阶乘函数prod，计算【例6.1.1-2】的1到20阶乘之和。

```
  s=0;n=1;
  while n<=20
      s=s+prod(1:n);
      n=n+1;
  end
  s
s =
  2.5613e+018
```

6.2.2 选择结构

if-end 选择结构

选择结构的一般形式是：

```
if 逻辑表达式1
    语句体1;
else
    语句体2;
end
```

else 部分表示：当计算的表达式结果为假时，就做 else 后面语句体 2 的工作。

【例 6.2.2-1】　比较 a 和 b 的大小。

```
a='a';b='b';
if (a<b)
  disp('true')
else
  disp('false')
end
```

if 语句嵌套

```
if 逻辑表达式 1
    语句体 1;
elseif 逻辑表达式 2
    语句体 2;
...
else
    语句体 else
end
```

需要注意：if 语句嵌套时，if 和 else 必须对应，否则容易出错。在 else 子句中也可嵌套 if 语句，这就形成了 elseif 结构。

【例 6.2.2-2】　选择灰色等级（如图 6.2.2-1 所示）。

(1)编写出函数文件 incolor.m。

```
function gd=incolor(colr)
  gd=[0 0 0];
if colr<3;
    gd=[.8 .8 .8];
elseif (colr>=3 ) &(colr<6);
    gd=[.6 .6 .6];
else
    gd=[.4 .4 .4];
end
```

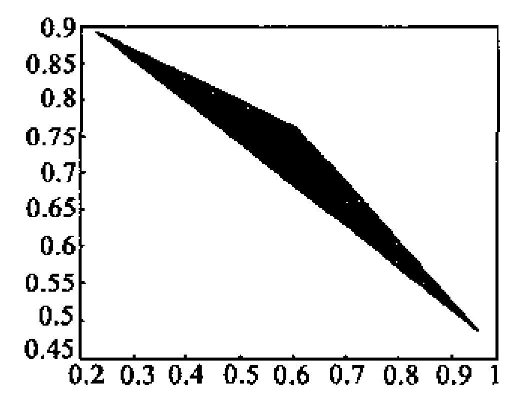


图 6.2.2-1　灰色等级选择

(2)在 MATLAB 命令窗运行。

```
incolor(9)
fill(rand(1,3),rand(1,3),ans);
```

switch 多分支选择结构

switch 语句和 if 语句类似，可根据变量或表达式的取值不同分别执行不同的

命令。其基本调用格式为：

```
switch  表达式
case  data 1
语句体 1；
case data 2
语句体 2；
...
otherwise
语句体 otherwise；
end
```

【例 6.2.2-3】 用菜单选择灰色等级(与图 6.2.2-1 相同)。

```
echo off
gd=[0 0 0];
m=menu('请选择灰色等级','浅灰','中灰','深灰') %menu 可对每行菜单项返回自
然数
switch  m
case 1
gd=[.8  .8  .8];
case 2
gd=[.6  .6  .6];
otherwise
gd=[.4  .4  .4];
end
fill(rand(1,3),rand(1,3),gd);
```

6.3 字 符 操 作

MATLAB 和 C 语言一样，也具有很丰富的字符操作功能，操作方法相似，甚至有些函数名都相同，它们被放在 matlab\toolbox\matlab\strfun 子目录下。

isstr	判断是否为字符
blanks	创建空白字符串
deblank	删去字符串结尾空白字符
eval	运行字符宏定义
strcmp	比较字符串

strtok	查找空格前的字符串
findstr	从一个字符串中查找是否包含另一个字符串
strrep	用一个字符串代替另一个字符串
upper	将字符串变为大写形式
lower	将字符串变为小写形式
real	将字符串转化为实数 ASCII 码
abs	将字符串变为 ASCII 码值
setstr	将 ASCII 码值变为字符串
num2str	将数字变为字符串
int2str	将整数变为字符串
str2num	将字符串变为数字
str2mat	将字符变为文本矩阵
sprintf	将带格式的数字转变为字符串
scanf	将字符串转变为带格式的数字
hex2num	将十六进制的字符串转变为 IEEE 浮点数
hex2dec	将十六进制的字符串转变为十进制数
dec2hex	将十进制的数转变为十六进制字符串

1. 字符串的产生

在 MATLAB 中所有字符串都用单引号界定后输入或赋值。

如命令 s='stu'的运行结果是：

```
s =
    stu
```

2. 字符矩阵

字符串的每个字符(空格也是字符)都是组成矩阵的一个元素，并占一列。如上述 s 变量是 1×3 的矩阵，可用命令 size(s)查得。生成矩阵时，每个字符串的长度必须相等。

【例 6.3-1】 字符矩阵生成。

```
a=['123' '456';'789' '233']
b=['123456';'789233'];
strcom(a,b)          %经比较 a 和 b 相同
```

```
a =
    123456
    789233
ans =
      1
```

3. 字符与 ASCII 的转换

MATLAB 以 16 位 ASCII 码储存每个字符。函数 double(或 abs)、char(或 setstr) 可以将字符和 ASCII 码互换。

【例 6.3-2】 字符和 ASCII 码互换。

```
a=double('abc ')
b=abs('abc ');   %a 和 b 相同
a =
   97 98 99 32
c=char([a])
c=
   abc
```

4. 字符与细胞矩阵的转换

实现转换的命令：

cellstr　　将细胞矩阵变成字符矩阵。

【例 6.3-3】 字符与细胞矩阵的转换。

```
a=['abc' 'def' 'ghi'];
b=cellstr(a)        %变成细胞
class(b)            %查看 b 类型
char(b)             %将细胞变成字符
b='abcdefghi'
ans =
        cell
ans =
        abcdefghi
```

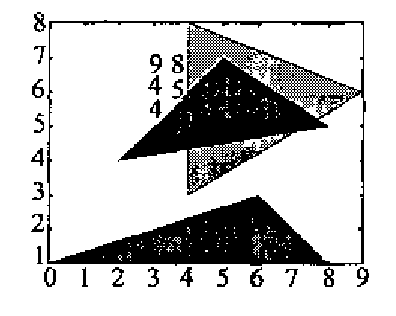


图 6.3-4　将随机阵用图形表现

5. 字符串和数值之间的互换

实现转换的命令：

int2str　　把整数换成字符。

num2str(x,n)　　把数值矩阵换成字符矩阵，x 为数值矩阵，n 为输出保留小数位数。

【例 6.3-4】 将随机阵画出，并把矩阵标注在图形上(如图 6.3-4)。

```
x=fix(10 * rand(3)) ;
```

```
y=fix(10*rand(3));
c=fix(10*rand(3));
s=num2str(x,3);
fill(x,y,c)
text(3.0,6.0,s)
```

6. 字符串的比较和替换

strcmp(str1,str2) 比较两个字符串是否相等。

strrep(old,s1,s2) 用s2替换old中的s1字符串。

【例6.3-4】 字符的替换。

```
rand('seed',0)
a=fix(10*rand(1,10))
s=num2str(a,3);       %数值阵转换成符号阵
strrep(s,'9','*')     %用'*'替换s阵中的'9'
strcmp(a,s)
a =
    2    0    6    6    9    3    5    8    0    0
ans =
    2    0    6    6    *    3    5    8    0    0
ans =
    0
```

7. 运行字符宏定义

在MATLAB中，可用字符建立一个宏定义，然后用eval函数运行。

【例6.3-5】 用eval函数计算2-4阶pascal矩阵。

```
for n = 2:4
eval(['P' num2str(n) ' = pascal(n)'])
end
```

eval的宏文字功能可以把函数名传入函数文件。构造字符串时应该注意方括号“[ ]”的使用。

6.4 结 构 体

与C语言相似，MATLAB语言也有结构体。在科学计算时，人们常常需要将一些类型不同的数据组合在一起，作为一个整体来处理。例如，学生的学习成绩档案、职工的人事档案、机械零件属性数据库等等。在MATLAB中，结构体的元素可以是数据型、字符型、细胞型，并可建成多维矩阵。

6.4.1　结构体的建立

比C语言简单，不需要在程序开始定义，一般用赋值的方法或用函数struct来定义。

【例6.4.1-1】　有一学生档案如下，建立结构体。

No	Name	Sex	Score
9862120	王　永	男	86　80　89　72　90
9862121	张丽丽	女	90　87　92　78　82

```
    student(1).name='王 永';
    student(1).num=9862120
    student(1).score=[86 80 89 72 90]
    student(2).name='张丽丽';
    student(2).num=9862121
    student(2).score=[90 87 92 78 82]
    或者:
    student(1) = struct('name','王  永','num','9862120','score',[86 80 89 72 90])
    student(2) = struct('name','张丽丽','num','9862121','score',[90 87 92 78 82])
student(1) =
    name: '王 永'
      num: '9862120'
    score: [86 80 89 72 90]
student(2)=
    name: '张丽丽'
      num: '9862121'
score: [90 87 92 78 82]
```

6.4.2　结构体的运算

对于结构体中变量的提取、重新赋值等可以用MATLAB的函数，也可直接操作。

(1)提取

F = getfield(S,'field')　　提取元素:S结构体,'field'元素。

F = getfield(S,{i,j},'field',{k})　　提取元素属性值:{i,j}元素下标;{k}属性下标。

【例 6.4.2-1】　提取【例 6.4.1-1】结构体的数据。

```
    F = getfield(student(2),'score')
    F2 = getfield(student,{2},'score',{2})
    f=student(2).score              %直接提取,与 F 相同
    f2=student(2).score(2)          %直接提取,与 F2 相同
F =
    90    87    92    78    82
F2 =
    87
```

(2) 重新赋值

S = SETFIELD(S,'field',V)　　　修改元素(V 元素)。

S = SETFIELD(S,{i,j},'field',{k},V)　　修改元素属性值(V 元素值)。

【例 6.4.2-2】　修改【例 6.4.1-1】结构体的数据。

```
    student = SETFIELD(student,{2},'score',{2},88)
    %与 student(2).score(2) = 88 等价
```

(3) 运算

有了结构体,进行整体运算操作非常方便,例如:

mean(student(2).score)　　求 student(2).score 的平均值。

mean([student.score])　　求所有 student.score 的平均值。

sum([student.score])　　等价于下面程序,求所有 student 中的 score 值。

```
    g=0
    for i=1:2
    g=g+sum([student(i).score]);
    end
```

6.5　数据输入输出

数据输入、输出在数值计算中是很重要的,在 MATLAB 中可选择的方法很多。

数据的输入:

(1)利用方括号按元素列表方式直接输入数据。

(2)用 M 文件按元素列表方式产生数据。

(3)从 ASCII 数据文件装载 load 数据。

(4)利用 fopen,fread 等低层 I/O 命令读数据。

(5)由内外部应用程序生成 MAT 文件。

(6)外部应用程序生成MEX 文件去读取数据。

(7)利用数据库传递数据。

数据的输出：

(1)利用diary 命令产生日记文件输出数据。

(2)利用notebook 获取数据。

(3)利用命令 save 输出数据。

(4)利用fopen,fwrite 等底层 I/O 命令输出数据。

(5)用外部应用程序生成MEX 文件写数据。

(6)用MAT 文件与外部应用程序进行数据交换。

由于篇幅所限，本节只讨论几种较重要的方法。

6.5.1 底层 I/O 命令的数据输入与输出

以下是MATLAB关于数据文件操作的函数，和C语言很相似。

fopen

打开文件进行读写。其格式如下：

FID=fopen('filename',PRECISION)

若打开成功，返回一整数，以后便代表此文件，否则返回－1。'filename'为文件名，读写控制方式 PRECISION 为下列字符之一：

'r'：　打开文件，读数据，文件必须存在。

'w'：　打开文件，写数据，文件不存在就建立新文件。

'a'：　打开文件，添加数据，文件不存在就建立新文件。

'r+'：　打开文件供读与写，文件必须存在。

'w+：　打开文件供读与写，文件不存在就建立新文件。

'a+'：　打开文件供读与添加，文件不存在就建立新文件。

上面格式都是二进制格式，如果想用ASCII格式，必须加上t字符。例如用'rt'代替'r'，用'wt+'代替'w+'。

fclose

关闭打开过的文件，其格式为：

fclose(FID)　　关闭FID文件。

若成功，返回0，否则返回－1。

fread

从二进制文件读数据。

格式一：[A,COUNT]=fread(FID,SIZE,PRECISION)

把数据文件里的二进制数据读到矩阵A里。COUNT为读到的数据元素个

数,FID是用fopen打开的文件标识。SIZE是一个读取数据选项,如果缺省,就读整个文件的内容,如果没缺省它的值可以是:

N:读取N个元素到一个列向量。

inf:读整个文件。

[M,N]:读数据到M×N大小的矩阵里去,数据按列顺序存放。N可以是无穷大inf,但是M不行。

PRECISION控制所读数据的格式,缺省为'uchar'格式,格式如下:

控制格式	说　明
'char'	字符,8bits
'schar'	符号字符,8bits
'short'	整数,16bits
'int'	整数,16bits或32bits
'long'	长整数,32bits或64bits
'float'	实数,32bits
'double'	长格式实数,64 bits
'uchar'	无符号字符,8 bits
'ushort'	无符号整数,16 bits
'uint'	无符号整数,16 or 32bits
'ulong'	无符号整数,32 bits
'char'	字符,8bits
'float32'	实数,32bits
'float64'	实数,64bits
'int8'	整数,8bits
'int16'	整数,16bits
'int32'	整数,32bits
'intN'	整数,Nbits
'uintN'	无符号整数,Nbits

【例6.5.1-1】 把test.dat文件的内容读出,并赋给f。

```
fid=fopen('test.dat','r');
f=fread(fid);
s=setstr(f);          %把正整数变成字符串
fclose(fid)
```

格式二：[A,COUNT]=fread(FID,SIZE,PRECISION,SKIP)

选项SKIP指定每读一个数据后，越过SKIP个字符，即如果现在的位置为N，那么下一个数据的位置应当是N+SKIP。

fwrite

以二进制格式对文件写数据。

格式一：COUNT=fwrite(FID,A,PRECISION)

把矩阵A里的内容往FID定义的文件写，数据按列的顺序往文件里写，COUNT是写进矩阵A里元素的个数。FID是由fopen返回的文件标识，1表示标准输出，2表示标准错误输出。

PRECISION同fread。

【例6.5.1-2】 以二进制格式往文件test.bin写数据。

```
A=fix(10*rand(5));
FID=fopen('test.bin','w')
[B,Cnt]=fwrite(FID,A,'int')
  fclose (FID)
```

如果把上面的文件读到B矩阵则：

```
FID=fopen('test.bin','r');
    [B,Cnt]=fread(FID,[5,inf],'int');
      fclose(FID)
B同A;
      Cnt=25
```

格式二：COUNT=fwrite(FID,A,PRECISION,SKIP)

选项SKIP指定每写一个数据后，就跳过SKIP个字符，即如果现在的位置为N，那么下一个数据的位置应当是N+SKIP。

fscanf

以选定的格式从文件读数据。格式如下：

[A,COUNT]=fscanf(FID,FORMAT,SIZE)

把FID标识的文件里的数据，以选定的输出格式赋给矩阵A。其中FID为文件标识，FORMAT为格式输出控制字符串，与C语言相似，但功能要强。可用%加上下面的一个字符表示不同格式输出：

d,i,o,u,x,e,f,g,s,c及[...]

e 为实数，科学计算法表示；f 为实数，小数表示；d 为整数格式；s 为字符格式；[...]表示越过。MATLAB 将会使读取数据自动循环下去，直到形成 A 阵为止。

例如：S=fscanf(FID,'%5d')

SIZE：可缺省，它控制矩阵 A 的形状与所读数据的个数。不缺省，可以选下面各种情况：

N：读取 N 个元素到一个列向量。

inf：读整个文件。

[M,N]：读数据到 M×N 大小的矩阵里去，数据按列顺序存放。M 为大于等于 1 的正整数，N 可以是无穷大 inf，但是 M 不行。

A：为返回的数据矩阵。

COUNT：可缺省，返回元素的个数。

fprintf

以选定的格式对文件写数据：

COUNT=fprintf(FID,FORMAT,A,…)

将 A 阵里的内容按选定格式对文本文件进行写数据，其格式同上。

【例 6.5.1-3】 将矩阵 *y* 的数据元素写到 sinexp.txt 中。

```
x=[0:0.1:1];y=[x;sin(x).*exp(x)];
FID=fopen('sinexp.txt','w');
fprintf(FID,'%6.2f    %12.4f\n',y);
fclose(FID)
```

```
0.00        0.0000
0.10        0.1103
0.20        0.2427
0.30        0.3989
0.40        0.5809
0.50        0.7904
0.60        1.0288
0.70        1.2973
0.80        1.5965
0.90        1.9267
1.00        2.2874
```

fgetl/fgets (FID)

无格式整行读。

fgetl 与 fgets 每次可读出文件中的一行，以回车键为结束标志。fgetl 与 fgets 的惟一区别是：fgetl 舍去行结束符，而 fgetl 则保留行结束符。

S=fgetl(FID)可读出标识为FID文件中的一行内容，然后指针下移一行。如果到了文件的结束位置，返回−1。

【例 6.5.1-4】 显示 test.m 文件的内容。

```
FID=fopen('test.m');
while 1
line = fgetl(FID);
if ~isstr(line), break, end
disp(line)
end
fclose(FID);
```

frewind

指针定位到开头。

frewind(FID)：把当前指针定位到文件的开头位置。

fseek

定位到特定位置。

STATUS = fseek (FID, OFFSET, ORIGIN) 把指针重新定位到相对于ORIGIN 的 OFFSET 位置。

其中 FID 为文件标识。

OFFSET 表示偏移方向：

>0 移向文件结尾；

=0 不改变位置；

<0 向文件头移动。

偏移基点 ORIGIN 的值可以是：

'bof'或−1：文件头开始；

'cof'或 0：当前位置开始；

'eof'或 1：文件末开始。

如果成功，STATUS 返回 0，否则返回−1。

例如 fseek(FID,0,−1)相当于 frewind 命令。

ftell

返回当前指针位置。

ftell 用于返回当前指针位置，格式为：POSITION=ftell(FID)。POSITION 返回的是相对文件开始处的距离，用字节计算。如失败就返回−1，用 FERROR 可查出错误性质。

feof

文件结束测试。

feof(FID)用于检查文件是否结束，如果结束，则返回 1，否则返回 0。

ferror

获取文件操作错误信息。

MESSAGE＝ferror(FID,'clear')返回文件 I/O 操作错误信息，'clear'是清除文件操作错误信息。省略此项，则表示不清除错误信息。MESSAGE 等于 0 为标准输入，1 为标准输出，2 为标准错误。

[MESSAGE,ERRUNM]＝ferror(FID)也可返回错误代码 ERRUNM。如果 MESSAGE 为空，ERRUNM 为 0。ERRUNM 错误值与 C 库返回的值是一致的。

6.5.2 MEX 动态连接函数接口

在 MATLAB 中，可以调用自己开发的 C 或 fortran 子程序，通过 MATLAB 的 API 函数库将 C 或 fortran 子程序编译成动态连接函数(库)，即 MEX 文件。在 WIN95 下，它是 32 位 DLL 格式的，可在 MATLAB 环境中直接调用或连接这些子程序，这样可提高速度，弥补 MATLAB 不足。例如：

(1)对于 A/D 或 D/A 卡，或其他 PC 硬件，可用 MEX 文件进行访问、采集数据、低级操作等。

(2)对于 MEX 文件，还可以使用其他资源，如 WINDOWS 的用户界面资源等。

(3)对于瓶颈计算，常常是 for 循环，M 文件运行速度没有 C 和 FORTRAN 那么快。

C 语言 MEX 文件工作原理

MEX 文件是由 MEX 源代码文件经过适当的编译器编译和链接器连接而生成的动态连接子程序。它定义被 MATLAB 调用的外部子程序的入口地址，定义 MATLAB 系统向子程序传递的子程序参数，还定义子程序向 MATLAB 系统返回的结果参数，以及调用计算功能子程序等。

入口子程序的函数名必须是 mexFunction，其构成：

```
void mexFunction(
int nlhs,mxArray *plhs[],
int nrhs, const mxArray *plhs[ ])
{
/*C 代码*/
}
```

下面对 mexFunction 函数中的参数具体说明：

nrhs　整数型变量，调用MEX文件的mxArray(或Matrix)类型输入参数的个数，MATLAB函数的右端(输入端)变量个数。

prhs　指针数组变量，其元素是指向MEX文件的mxArray(或Matrix)类型输入参数的变量指针。

nlhs　整数型变量，调用MEX文件的mxArray(或Matrix)类型输入参数的个数，MATLAB函数的左端(输出端)变量个数。

plhs　指针数组变量，其元素是指向MEX文件的mxArray(或Matrix)类型输入参数的变量指针。

调用形式：[a,b,c,...]=fun(d,e,f,...)

a,b,c　MATLAB函数的左端(输出端)变量。

d,e,f　MATLAB函数的右端(输入端)变量。

【例6.5.2-1】 编一个C程序，输入一个数，然后乘2。

```
#include "mex.h"
/* timestwo 子程序 */
void timestwo(double y[], double x[])
{
y[0] = 2.0 * x[0];
}

void mexFunction( int nlhs, mxArray *plhs[],
int nrhs, const mxArray *prhs[] )
{
double *x, *y;
int mrows,ncols;
/* 核查变量数目 */
if(nrhs! =1) {
mexErrMsgTxt("One input required.");
}
else if(nlhs>1) {
mexErrMsgTxt("Too many output arguments");
}
/* 输入必须是非复数向量 */
mrows = mxGetM(prhs[0]);
ncols = mxGetN(prhs[0]);
if( ! mxIsDouble(prhs[0]) || mxIsComplex(prhs[0]) ||
! (mrows==1 && ncols==1) ) {
```

```
mexErrMsgTxt("输入必须是非复数向量.");
}
/* 生成矩阵 */
plhs[0] = mxCreateDoubleMatrix(mrows,ncols, mxREAL);
/* 将指针指向每个输出输入矩阵 */
y=mxGetPr(plhs[0]);
x=mxGetPr(prhs[0]);
/* 调 timestwo 子程序 */
timetwo(y,x)
}
```

编译连接运行。

C 和 FORTRAN 程序的编译连接

(1)设置编译环境

① 进入 MS-DOS 环境。

② 进入 matlab\bin 子目录。

③ 键入 mex-setup。

④ 选择编译器。

出现菜单后,根据情况进行选择。假如,在你的计算机 d:\vc5 里已装了 Microsoft Visual C++5.0,则可输入:

```
Welcome to the utility for setting up Compilers for building external interface files
C Compilers:
[1] Microsoft Visual C++
[2] Borland C/C++
[3] Watcom C/C++
Fortran Compilers:
[4] Microsoft PowerStation
[0] None
Compiler:1
```

版本选择:

```
which version:
[1] Microsoft Visual C++ 4.x
[2] Microsoft Visual C++ 5.x
version:2
Please enter to location of your C compiler:[c:\msdev]:d:\vc5
```

验证:

```
Please Verify
Compiler:Microsoft Visual C++ 5.x
Location:d:\vc5
ok? ([y]/n):y
```

直到 mexopts.bat 文件生成：

```
'mexopts.bat' is being updated
```

(2)编译连接 timetwo.c。

在 DOS 或 MATLAB 环境下键入：

```
mex   timetwo.c
或   cmex   timetwo.c
```

稍等片刻产生 timetwo.def，timetwo.map，timetwo.dll(或 timetwo.mex)文件。

(3) 运行

在 MATLAB 环境下键入：

```
  x = 2;
  y = timestwo(x)
y =
    4
```

6.6 图形用戶界面 GUI 的设计

6.6.1 菜单对象的创建

菜单对象的创建函数是 uimenu，可以用于创建菜单条、下拉菜单及其菜单项。其格式为：

hmenu1=uimenu(h,'属性名',属性值,…)

这里，h 为图形窗口句柄，如果此项缺省，就在当前窗口生成菜单；如果没有当前窗口，MATLAB 就自动打开一个图形窗口，并在它下面生成菜单。hmenu1 为生成的菜单句柄。

生成菜单项或子菜单对象的格式为：

hsubmenu1=uimenu(hmenu1,'属性名',属性值,…)

hmenu1 为上一级菜单生成的菜单句柄；hsubmenu1 为本级菜单生成的菜单句柄。

菜单常用属性：

1. 公共属性

(1) Children 属性:取值为空矩阵或句柄。

(2) Parent 属性:取值为矩阵,表明父级菜单或菜单所在的图形窗口。

(3) Tag 属性:取值为字符串,为该菜单的标识,其他程序通过此标识可找到该菜单。

(4) Type 属性:取值是 uimenu,可标明图形对象的类型。

(5) UserData 属性:取值是一个矩阵,为应用程序传递数据。

(6) Visible 属性:取值可以是 on 或 off(缺省),已确定该对象的可见性。

2. 基本控制属性

(1) Callback 属性

用户用鼠标在控制对象上操作时,将会使系统响应并运行 Callback 属性定义的子程序或宏命令(Callback 调用)。

CallBack 的值是字符类型,它可以是一个 M 文件的名字,也可以是一个 MATLAB 语句组成的字符串,而 MATLAB 语句中自身的函数或命令必须用双引号括起来。

(2) Accelerator 属性

Accelerator 属性值是字符类型,只对子菜单 Children 属性值为空的对象,它可定义该菜单项的热键。用 Ctrl 加该字符,即可激活该菜单项的功能。

(3) Checked 属性

Checked 属性值是 on 或 off(缺省),它可为某菜单项标记选择状态。

例如:
```
solid=uimenu(Hm-lstyle,'Label','Solid',...
        'CallBack',['ltype="-";',...
        'set(solid,"checked","on"),',...
        'set(dotted,"checked","off"),',...
        'set(dashed,"checked","off")']);
```

(4) Enable 属性

Enable 属性值是 on 或 off(缺省),它控制菜单的可选择性。如果它的值是 off,不能使用该菜单项。此时该菜单项为暗色。

(5) Label 属性

Label 属性值是字符串,定义菜单项的名字,可在其中加上"&"字符,后面的字符便有一条下划线,使用户用 Alt 键加上该字符来激活该菜单项。

(6) Position 属性

Position 属性值是整数,可定义主菜单和菜单项在菜单中的相对位置。

(7) Separator 属性

Separator 属性值是 on 或 off(缺省),如果该值为 on,则在该菜单项上方加一

根横线。

【例6.6.1-1】 菜单的建立。

(1)主菜单项建立

```
Hm-line = uimenu(gcf,'label','Line');
```

(2)一级下拉菜单建立

```
Hm-lstyle = uimenu(Hm-line,'label','Line Style');
Hm-lwidth = uimenu(Hm-line,'label','Line Width');
Hm-color = uimenu(Hm-line,'label','Line Color');
```

(3)Hm-lstyle 下面的二级菜单建立

```
uimenu(Hm-lstyle,'Label','Solid',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineStyle'',''-''),',...
        'end']);
uimenu(Hm-lstyle,'Label','Dotted',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineStyle'','':''),',...
        'end']);
uimenu(Hm-lstyle,'Label','Dashed',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineStyle'',''--''),',...
        'end']);
uimenu(Hm-lstyle,'Label','DashDot',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineStyle'',''-.''),',...
        'end']);
```

(4)Hm-lwidth 下面的二级菜单建立

```
uimenu(Hm-lwidth,'Label','Default',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineWidth'',0.5),',...
        'end']);
uimenu(Hm-lwidth,'Label','Thick',...
    'CallBack',['waitforbuttonpress;',...
```

```
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineWidth'',2.0),',...
        'end']);
uimenu(Hm-lwidth,'Label','Thicker',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineWidth'',3.0),',...
        'end']);
uimenu(Hm-lwidth,'Label','Thickest',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''LineWidth'',4.0),',...
        'end']);
```

(5)Hm-lcolor 下面的二级菜单建立

```
uimenu(Hm-lcolor,'Label','Yellow',...
    'BackgroundColor','y',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''y''),',...
        'end']);
uimenu(Hm-lcolor,'Label','Magenta',...
    'BackgroundColor','m','ForegroundColor','w',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''m''),',...
        'end']);
uimenu(Hm-lcolor,'Label','Cyan',...
    'BackgroundColor','c',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''c''),',...
        'end']);
uimenu(Hm-lcolor,'Label','Red',...
    'BackgroundColor','r','ForegroundColor','w',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''r''),',...
```

```
        'end']);
uimenu(Hm-lcolor,'Label','Green',...
    'BackgroundColor','g',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''g''),',...
        'end']);
uimenu(Hm-lcolor,'Label','Blue',...
    'BackgroundColor','b','ForegroundColor','w',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''b''),',...
        'end']);
uimenu(Hm-lcolor,'Label','White',...
    'BackgroundColor','w',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''w''),',...
        'end']);
uimenu(Hm-lcolor,'Label','Black',...
    'BackgroundColor','k','ForegroundColor','w',...
    'CallBack',['waitforbuttonpress;',...
        'if get(gco,''Type'') == ''line'',',...
            'set(gco,''Color'',''k''),',...
        'end']);
```

(6)第二主菜单项建立

```
Hm-cmap = uimenu(gcf,'Label','Color Map');
```

(7)Hm-cmap 下面的子菜单项建立

```
uimenu(Hm-cmap,'Label','Lighter','CallBack',
        'brighten(.3)');
uimenu(Hm-cmap,'Label','Darker','CallBack',
        'brighten(-.3)');
uimenu(Hm-cmap,'Label','Default','CallBack',
        'colormap(''default'')');
uimenu(Hm-cmap,'Label','Gray','CallBack',
        'colormap(gray)');
uimenu(Hm-cmap,'Label','Hot','CallBack',
```

```
        'colormap(hot)');
uimenu(Hm-cmap,'Label','Cool','CallBack',
        'colormap(cool)');
uimenu(Hm-cmap,'Label','Bone','CallBack',
        'colormap(bone)');
uimenu(Hm-cmap,'Label','Copper','CallBack',
        'colormap(copper)');
uimenu(Hm-cmap,'Label','Pink','CallBack',
        colormap(pink)');
uimenu(Hm-cmap,'Label','Prism','CallBack',
        'colormap(prism)');
uimenu(Hm-cmap,'Label','Jet','CallBack',
        'colormap(jet)');
uimenu(Hm-cmap,'Label','Flag','CallBack',
        'colormap(flag)');
uimenu(Hm-cmap,'Label','HSV','CallBack',
        'colormap(hsv)');
```

(8)第三主菜单项建立

```
Hm-quit = uimenu(gcf,'Label','Quit');
```

(9)Hm-quit 下面的子菜单项建立

```
uimenu(Hm-quit,'Label','Close Figure','CallBack','close; return');
uimenu (Hm-quit,'Label','Remove  Menus','CallBack', ['delete (findobj (gcf,''Type'',''
uimenu'',''Parent'',gcf)),','drawnow']);
```

6.6.2 控制子窗创建

控制子窗对象包括按钮、收音机按钮、滚动条、弹出式菜单、编辑框、边框窗口等等，它的创建函数是 uicontrol。其格式为：

hctr1=uicontrol(h,'属性名',属性值,…)；

这里，h 为图形窗口句柄，如果此项缺省，就在当前窗口生成此菜单；如果没有此窗口，MATLAB 就自动打开一个图形窗口，并在它下面生成控制子窗对象。

控制子窗属性

1. 公共属性

(1)Children 属性：取值为空矩阵。

(2)Parent 属性：取值为图形窗口句柄值，因为其父辈总是图形窗口。

(3)Tag 属性：取值为字符串，为该菜单的标识，其他程序通过此标识可找到该子窗。

(4)Type 属性:取值是 uicontrol,可标明图形对象的类型。

(5)UserData 属性:取值是一个矩阵,为应用程序传递数据。

(6)Visible 属性:取值可是 on 或 off(缺省),以确定该对象的可见性。

2. 基本控制属性

(1)BackgroundColor 属性:取值为颜色预定义字符或 RGB 数值,以定义其背景色。

(2)Callback 属性:取值是字符串,可以是 M 文件名或一小段 MATLAB 语句。

(3)Enable 属性:取值为 on(缺省值)、off 和 inactive。

(4)Extent 属性:取值为[0,0,width,height],记录子窗标题字符的位置大小。

(5)ForegoundColor 属性:取值为颜色预定义字符或[RGB]值。

(6)Max、Min 属性:取值是数值,其缺省值分别是 1 或 0。

(7)String 属性:取值是字符串矩阵或块数组,它可定义子窗的标题或选项。

(8)Style 属性:取值为子窗名称,如 pushbutton,radiobutton,checkbox,edit,slider,frame 等。

(9)Units 属性:取值可以是 pixels(缺省),normalized,inches,centimeters,points。

(10)Value 属性:取值是向量值,也可以是数值。

一、按钮的创建

下面的函数可生成子窗控制,通过 Style 的属性值 pushbutton 设置,建立按钮。Position 的属性值[40 20 100 25]分别是 x 坐标、y 坐标、按钮长度、按钮高度。Callback 属性定义了按下按钮后所运行的应用程序。

```
pbstart=uicontrol(gcf,'Style','pushbutton',...
        'Position',[40 20 100 25],...
        'String','开始','CallBack','drum1');
```

二、文本控制

```
txt=uicontrol(gcf,'Style','text','Position',
     [1 44 100 20],...
       'String','文本显示');
```

三、检测框创建

前两项属性值,与前面相似。String 可定义检测框的标题,Callback 可将检测框的缺省值设为 off,如果点击检测框后,就变为 on。

```
bbox=uicontrol(gcf,'Style','checkbox','Position',[ 130 44 100 20],...
   'String','检测框','CallBack',['set(gca,"Box","off"),',...
   'if get(bbox,"Value")==1,', 'set(gca,"Box","on"),',...
```

```
    'end']);
```

四、无线电按钮的创建

此例说明如果用户按下按钮后，就将 rbsys 变量设置为 0，否则设置为 1。

```
rbstartat=uicontrol(gcf,'Style','radio','String',...
    'Start','Position',...
    [160 180 80 25],'CallBack',['if get(rbstatat,"Value")==1),',...
    'set(rbsys,"Value",0),',...
    'else set(rbsys,"Value",1),',...
    'end']);
```

五、滑动条的创建

功能：当用户移动滑动条，就将当前值写在滑动条名称的右侧，并对 View 进行设置。

```
slid=uicontrol(gcf,'Style','slider',...
        'Position',[50 50 120 20],'Min',-90,...
'Max',90,'Value',30,'CallBack',
        ['set(scur,"String",',...
        'num2str(get(slid,"Value"))),',...
        'set(gca,"View",',...
        '[get(slid,"Value"),',...
        'get(sliev,"Value")]')];
min=uicontrol(gcf,'Style','text',...   %将 slid 中设置的 Min 值写在滑动条的左侧
        'Position',[20 50 30 20],'String',...
        num2str(get(slid,'Min')));
max=uicontrol(gcf,'Style','text',...   %将 slid 中设置的 Max 值写在滑动条的右侧
        'Position',[170 50 30 20],'String',...
        num2str(get(slid,'Max')),...
        'HorizontalAlignment','right');
label=uicontrol(gcf,'Style','text',...   %滑动条名称
        'Position',[50 80 65 20],'String','滑动条示例');
scur=uicontrol(gcf,'Style','text',...
%将 slid 中设置的 Value 值写在滑动条名称的右侧
        'Position',[120 80 50 20],'String',...
        num2str(get(slid,'Value')));
```

六、弹出式菜单的创建

菜单的可选项只需在 String 属性中设置，每项之间用竖线字符“|”隔开，并用

单引号将所有的选项括起来。

Value属性中的值是弹出式菜单列表中选项的序号。如果用户选列表中第二项，则Value的属性便是－4。

下面的菜单列表中包含一组可供选择的地点，而且每种选择都与一个数值相对应。当某个地点被选中时，CallBack就按地点设置时间。

```
txtpop = uicontrol (gcf,'Style','text','Position',[10 180 120 20],'String',...'Show Local Time');
popcity=uicontrol(gcf,'Style','popupmenu',...
        'String','GMT|Boston|Paris|Beijing',...
        'Position',[10 155 120 25],...
        'UserData',[0;-4;1;8],...
        'CallBack',['ud=get(popcity,"UserData");',...
        'setclk(ud(get(popcity,"Value")))']);
```

七、编辑框的创建

创建一个多行的编辑框，由6行文字组成，此时Max属性可以是任何大于1的整数值，这是由于属性Min的缺省值是0。属性Max与属性Min之差小于或等于1时，为单位编辑框。

```
ed=uicontrol(gcf,'Style','edit','String','编辑框|可多行|编辑|文本',...
'Position',[10 200 75 100],'Max',6);
```

八、窗口边框的创建

需要建立窗口边框时，必须在建立子窗控制之前建立边框，否则边框会覆盖该组中其他的子窗对象。为了留出边框边界，边框所占用的区域要比该组中所有的控制对象占有的区域要大一些。

```
winframe=uicontrol(gcf,'Style','frame',...
        'Position',[195 15 110 110]);
```

【例6.6.2-1】 GUI界面的设计。

```
%按钮的设计
pbstart=uicontrol(gcf,'Style','push',...
        'String','Start Clock',...
        'Position',[10 10 120 25],...
        'CallBack','');
%关闭窗口按钮设计
pbclose=uicontrol(gcf,'Style','push',...
        'String','close windows',...
        'position',[160 10 120 25],...
```

```
        'callBack','close(gcf)');
%分组边框
framcopt=uicontrol(gcf,'Style','frame','Position',[ 5 55 130 80]...
'String','Auto Ticks');
%文本Options显示
txtopts=uicontrol(gcf,'Style','text','Position',
[ 10 110 120 20],...
'String','Options');
%复选框
cktick=uicontrol(gcf,'Style','checkbox','Position',
[ 10 85 120 25],...
'String','Auto Ticks');
cksecs=uicontrol(gcf,'Style','checkbox','Position',
[ 10 60 120 25],...
'String','Show Second');
%无线电按钮
rbstartat=uicontrol(gcf,'Style','radio','String',...
'Start at','Position',...
[160 180 80 25],'CallBack',['if
get(rbstatat,"Value")==1),',...
'set(rbsystime,"Value",0),',...
'else set(rbsystime,"Value",1),',...
'end']);
rbsystime=uicontrol(gcf,'Style','radio','String',...
'System time','Position',...
[160 155 120 25],...
'Value',1,'CallBack',
['if
get(rbstatat,"Value")==1),',...
        'set(rbsystime,"Value",0),',...
'else
set(rbsystime,"Value",1),',...
        'end']);
%滚动条
txtsh=uicontrol(gcf,'Style','text','Position',
[160 100 120 20],'String',... 'Ticker Volume');
```

```
txtvol=uicontrol(gcf,'Style','text','Position',
[160 80 120 20],'String',...'Soft...load ');
slivol=uicontrol(gcf,'Style','slider','Position',
[160 55 120 25],... 'Value',0.5) ;
%弹出菜单
txtpop=uicontrol(gcf,'Style','text','Position',
[10 180 120 20],'String',...'Show Local Time');
popcity=uicontrol(gcf,'Style','popupmenu',...
        'String','GMT|Boston|Paris|Beijing',...
        'Position',[10 155 120 25],...
        'UserData',[0;-4;1;8],...
        'CallBack',['ud=get(popcity,"UserData");',...
        'setclk(ud(get(popcity,"Value")))']);
%编辑框
edtime=uicontrol(gcf,'Style','edit',...
        'Position',[240 180 40 25],...
        'CallBack',['if
str2num(get(entime,"string"))<=0,',...
        'disp("Error-Value must be positive"),',...
        'end']);
% 菜单
Opts= uimenu(gcf,'label','option');
% 菜单下的子菜单
CType= uimenu(Opts,'label','Clock Types');
uimenu(CType,'Label','Digital','CallBack','ctype=1;')
uimenu(CType,'Label','Digital','CallBack','ctype=1;');
% 子菜单
settings=uimenu(Opts,'label','Setting',
        'Separator','on');
uimenu(Opts,'label','Set Start
        Time','Separator','on','CallBack','setstart');
% 菜单
Run= uimenu(gcf,'label','Run');
        uimenu(Run,'label','Start Clock',...
    'CallBack',' ');
LocTime=uimenu(Run,'label','Local
```

```
    time','Separator','on');
    uimenu(LocTime,'Label','GMT','CallBack',
            'adjtime(0)')
    uimenu(LocTime,'Label','Boston','CallBack',
            'adjtime(-4)');
uimenu(LocTime,'Label','Paris','CallBack',
    'adjtime(1)');
    uimenu(LocTime,'Label','Beijing','CallBack',
    'adjtime(8)');
    uimenu(Run,'Label','CloseClock','Separator',
    'on',...
    'CallBack','close(gcf)');
```

6.6.3 GUI 工具箱 Guide

与 windows 的开发工具 SDK 类似，MATLAB 也带了一个 GUI 设计工具——Guide。其中有 Guide 控制板（命令 guide）、属性编辑器（命令 propedit）、Callback 编辑器(cbedit)、菜单编辑器(menuedit)、位置调整工具(align)，但 Guide 是集成环境，启动时携带了后四种工具，如果你有 BC、VB 或 VC 的开发经验，这些将变得异常简单。

一、Guide 控制板

整个 Guide 主要有三大部分，如图 6.6.3-1 所示：

(1) Guide 工具

位于 Guide 控制板顶部，它包括属性编辑器、Callback 编辑器、位置调整工具、菜单编辑器。只要点击不同的图标按钮，便可打开相应的工具，使用方法后面再作介绍。

(2) Guide 控制的图形窗口列表

位于 Guide 控制板中部，可对打开的图形窗口进行管理和操作，共有两个选择：1. 控制的(Controlled)，在控期间由 Guide 控制板控制，没有独立操作权；2. 活动的(Active)，用户能够独立自由操作。

(3) 控制子窗建立工具箱

位于 Guide 控制板底部，含有按钮等 11 个控制子窗设计工具。设计时，按一下按钮，然后到相应图形窗口拉出矩形框，给出子窗的位置和大小。

二、属性编辑器

用于对控制子窗及图形属性的编辑，相当于命令行中的 set 和 get。在结构上

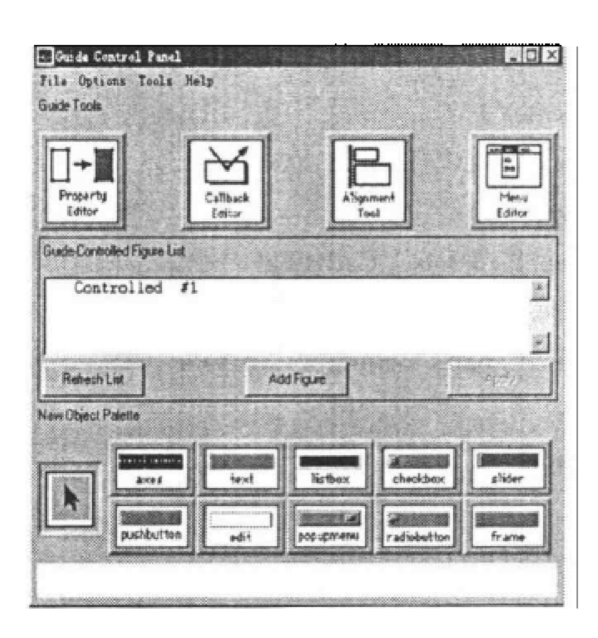


图 6.6.3-1　Guide 控制板

属性编辑器也分三部分，如图 6.6.3-2 所示：上部为图形对象列表框，在这里可以

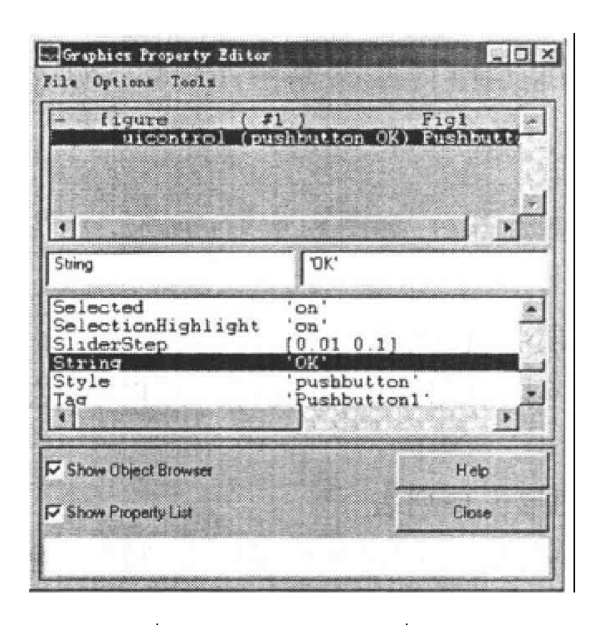


图 6.6.3-2　属性编辑器

进行图形窗口对象及控制子窗对象的选择；下部为属性列表框，显示图形窗口内全部对象属性和属性值；中部为属性编辑窗，编辑要选择的某对象属性值。

使用方法：

1. 如果在图形窗口中，已设计了一个按钮，在按钮上连击两次，等出现属性编

辑器后，首先在窗口上部，选择所要编辑的选项即 Uicontrol (Pushbutton)，然后在下面的属性列表框内找到选项如 String ，最后在中间的编辑窗内加以修改。

2. 在命令窗口中，可以键入 propedit(gcf)，也可调出属性编辑器，编辑方法如上。

三、Callback 编辑器

Callback 编辑器可以编辑某个图形对象的 Callback 属性值，而且可以省略 Callback 代码中所需的单引号。Callback 编辑器如图 6. 6. 3-3 所示，和属性编辑器

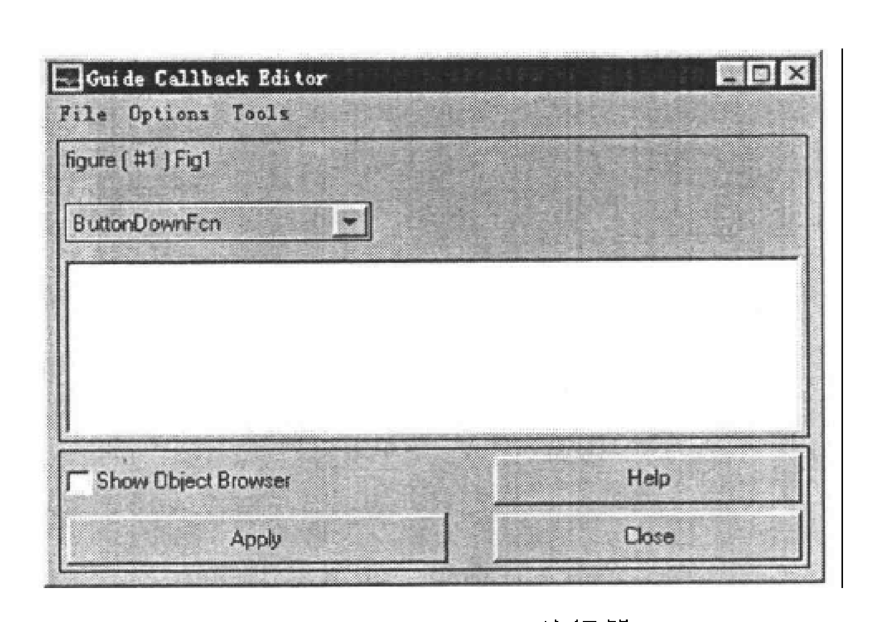


图 6. 6. 3-3　Callback 编辑器

相似也有一个图形对象列表框，如果选择了某个选项，便可以输入或编辑 Callback 代码，然后按一下 Apply 按钮以完成操作。

四、菜单编辑器

菜单编辑器如图 6. 6. 3-4 所示，编辑或设计用户菜单。但是，必须在图形窗口变为活动窗口之后，即在存盘和重新运行你所建立的 m 文件之后才能在图形窗口上显示所设计的菜单。注意增加菜单时需按 New Menu 按钮，然后分别用上下左右四个箭头按钮，来调整它们之间主次的关系。

五、位置调整工具

位置调整工具如图 6. 6. 3-5 所示，主要用于对两个以上的控制于窗对象进行位置之间的调整。那么首先必须在图形对象列表框上，用 shift 键或 ctrl 键加上鼠标点击进行选择对象，这时，图标由暗色变成白色后，再按下所需要的图标按钮，最后再按 Apply 按钮，立刻会看到被选择的对象做出了相应的位置变化。

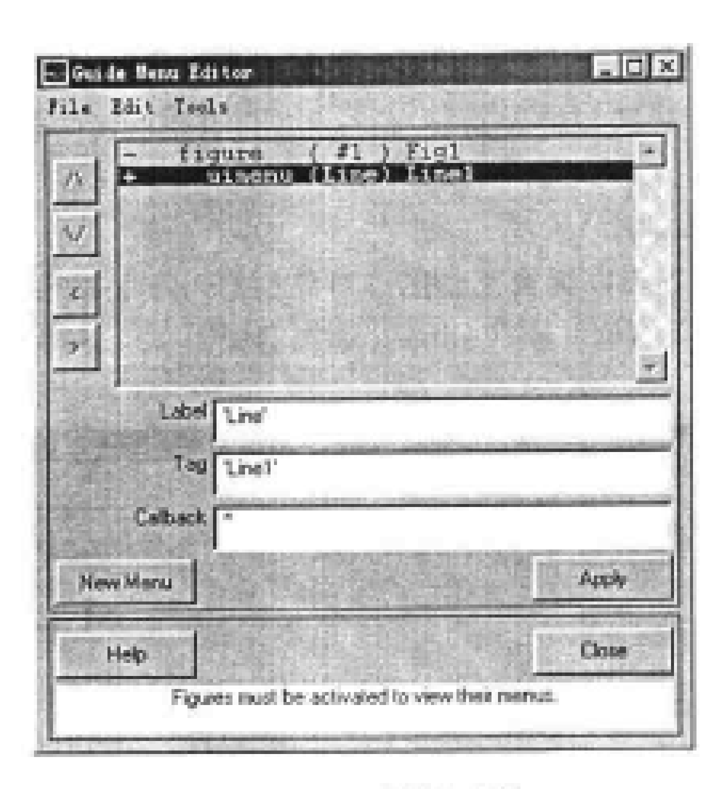


图 6.6.3-4 菜单编辑器

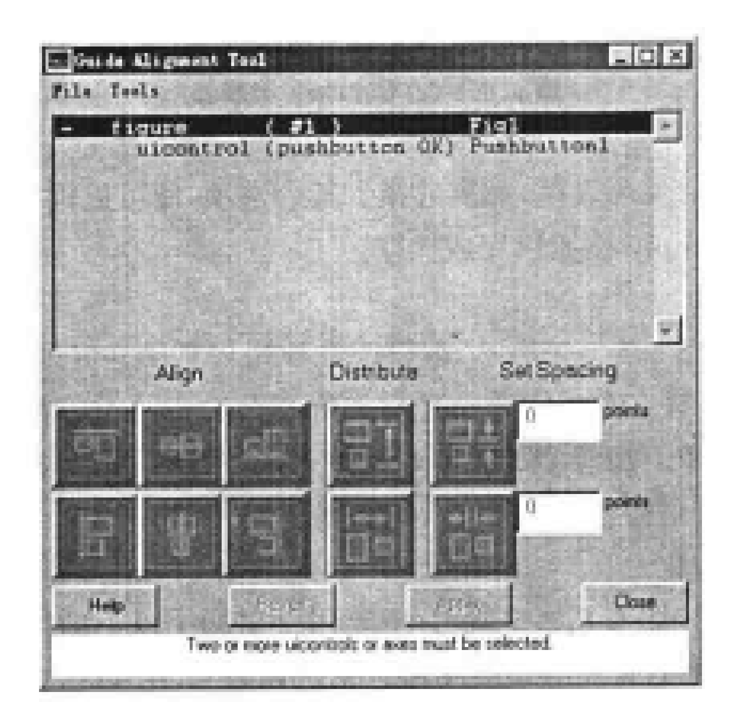


图 6.6.3-5 位置调整编辑器

习　题　六

1. 使用for,sum和prod命令,完成下列各项要求。

(1) 利用prod命令生成与列写的N的阶乘($N!$),并且对应$N=1\sim10$进行验证。

(2) 使用for命令代替prod命令,完成(1)项中的功能。

(3) 使用sum命令求与列写N个数的连续和,对应$N=1\sim10$进行验证。

(4) 使用for命令代替sum命令,完成(3)项中的功能。

(5) 编写一个程序,生成与列写第一个整数平方和,并且对应$N=1\sim10$进行验证。

2. 编写一函数程序,完成同时计算几个阶跃响应特征参数的功能。注:阶跃响应数据是一列矢量,与此同时对应几个响应,因此响应数据是一个矩阵(每一列对应不同的阶跃响应数据)。

3. 完成下列程序设计,比较各个系统的阶跃响应。

(1)求出下列传递函数的极点。利用第八章中的程序计算下列习题的阶跃响应特征参数($\sigma\%$,T_R、T_S,T_P)。

$$T_1=\frac{2}{S^2+2S+2},T_2=\frac{4S+2}{S^2+2S+2},T_3=\frac{1}{2S^3+3S^2+3S+1}$$

(2)编制一个程序,求系统的频率响应(ω,mag,phase),并求得输出频率特性(M_R、B_W)。

(3)使用上述程序,制表列写上述数据。

(4)比较加入零点(如T_2)与加入极点(如T_3)时时域与频域的结果。

4. 编制一个程序,该程序取传递函数的分子与分母作为程序的输入参数,然后求它的极点、独立的复数主导极点,并且计算它的阶跃响应特征参数。注:用该程序计算阶跃响应特征参数时,由于是利用主导极点与计算公式,因此与上述数值方法的结果有所区别。

5. 编写一个程序,将【例6.4.1-1】的结构体数据写入一个文件stu.dat,然后再从文件stu.dat中读出并显示到屏幕上。

6. 用FORTRAN或C语言编写一MEX程序,要求键盘输入【例6.4.1-1】的结构体数据后,按学生成绩进行排序。

7. 分别用编程的方法和集成环境Guide,设计一个具有菜单和控制窗口的小系统,该系统能够完成下面任务之一:

(1)交互地绘制专业工程图。

(2)控制系统动态仿真图形。

(3)解常微分方程动态仿真,可参照dde程序。

第七章　工程中偏微分方程的解法

7.1　偏微分方程的基本知识

在工程中，描述流体运动和场的分布的大多数数学模型是根据质量连续和能量守恒等基本原理推导出来的微分方程，加上适当的初始条件和边界条件所构成。在一维的问题中，它们是常微分方程；在二维或三维的问题中，它们是偏微分方程。

通常这些方程可取线性或非线性、齐次或非齐次的二阶偏微分方程。

1. 方程类型

PDE Toolbox 求解的基本方程有椭圆型方程、抛物型方程、双曲型方程、特征值方程、椭圆型方程组以及非线性椭圆型方程。

椭圆型方程：　$-\nabla\cdot(cu)+au=f,\quad in\ \Omega$

其中 Ω 是平面有界区域，c、a、f 以及未知函数 u 是定义在 Ω 上的实（或复）的函数。

抛物型方程：$d\frac{\partial u}{\partial t}-\nabla\cdot(c\nabla u)+au=f,\quad in\ \Omega$

双曲型方程：$d\frac{\partial^2 u}{\partial t^2}-\nabla\cdot(cu)+au=f,\quad in\ \Omega$

特征值方程：$-\nabla\cdot(c\nabla u)+au=\lambda du,\quad in\ \Omega$

其中 d 是定义在 Ω 上的复函数，λ 是求的特征值。在抛物型方程和双曲型方程中，系数 c、a、f 和 d 可以依赖于时间 t。

可以求解非线性椭圆型方程：

$$-\nabla\cdot(c(u)\nabla u)+a(u)u=f(u),\quad in\ \Omega$$

其中 c、a 和 f 可以是解 u 的函数。还可以求解如下 PDE 方程组：

$$-\nabla\cdot(c_{11}\nabla u_1)-\nabla\cdot(c_{12}\nabla u_2)+a_{11}u_1+a_{12}u_2=f_1$$

$$-\nabla\cdot(c_{21}\nabla u_1)-\nabla\cdot(c_{22}\nabla u_2)+a_{21}u_1+a_{22}u_2=f_2$$

利用命令行可以求解高阶方程组。对于椭圆型方程，可以用自适应网格算法，还能与非线性解结合起来使用。

另外，对于 Poisson 方程还有一个矩形网格的快速求解器。

2. 边界条件

(1)Dirichlet 条件

$$hu=r$$

(2)Neumann 条件

$$\boldsymbol{n}\cdot(c\nabla u)+qu=g$$

其中 $\boldsymbol{n}$ 是 $\partial\Omega$ 上的单位外法向矢量，g，q，h 和 r 是定义在 $\partial\Omega$ 上的函数。对于特征值问题仅限于齐次条件：g=0，r=0。对于非线性情形，系数 g，q，h 和 r 可以依赖于 u，对于抛物型方程和双曲型方程系数可以依赖于时间 t。对于方程组情形，Dirichlet 边界条件为

$$h_{11}u_1+h_{12}u_2=r_1 \qquad h_{21}u_1+h_{22}=r_2$$

而一般的 Neumann 条件为

$$\boldsymbol{n}\cdot(c_{11}\nabla u_1)+\boldsymbol{n}\cdot(c_{12}\nabla u_2)+q_{11}u_1+q_{12}u_2=g_1$$

$$\boldsymbol{n}\cdot(c_{21}\nabla u_1)+\boldsymbol{n}\cdot(c_{22}\nabla u_2)+q_{21}u_1+q_{22}u_2=g_2$$

混合边界条件为

$$h_{11}u_1+h_{12}u_2=r_1$$

$$n\cdot(c_{11}\nabla u_1)+n\cdot(c_{12}\nabla u_2)+q_{11}u_1+q_{12}u_2=g_1+h_{11}\mu$$

$$n\cdot(c_{21}\nabla u_1)+n\cdot(c_{22}\nabla u_2)+q_{21}u_1+q_{22}u_2=g_2+h_{12}\mu$$

其中 μ 的计算要使得 Dirichlet 条件满足。在有限元方法中，Dirichlet 条件也称为本质边界条件，Neumann 条件也称为自然边界条件。

3. PDE 模型的背景

PDEToolbox 中所解的 PDE 模型有着广泛的背景，它来自工程和科学的许多分支。

椭圆型和抛物型方程来自：

- 定常和非定常传输问题；
- 多孔介质的流动和扩散问题；
- 绝缘和导体材料的静电场问题；
- 势流。

双曲型方程来自：

- 暂态和谐波在声音和电磁场中的传播；
- 薄膜的横振动。

特征值问题来自求解薄膜和结构力学的固有振动问题。

4. 定解问题的设置

最简单的办法是在 PDETOOL 上直接使用图形用户界面(GUI)。设置定解问题分别采用下面三个模式(Mode)：

● Draw 模式：使用CSG 对话框建立几何模型，可选的工具有矩形、圆、椭圆和多边形；

● Boundary 模式：在各个边界段上给出边界条件；

● PDE 模式：确定方程的类型、系数 c，a，f 和 d，也能够在不同子区域设置不同的系数（反映材料的性质）。

5. 解 PDE 问题

GUI 解 PDE 问题主要使用两个模式：

● Mesh 模式：网格生成，自动控制网格参数；

● Solve 模式：对于椭圆型方程还能求非线性和自适应解。对于抛物型和双曲型方程，设置初边值条件后能求出给定 t 时刻的解。对于特征值问题，能求出给定区间内的特征值。求解后可以加密网格再求解。

6. 计算结果的可视化

从GUI 能够使用Plot 模式实现可视化。可以使用color，height 和 vector 控制参数作图。对于抛物型和双曲型方程，还可以生成解的动画。这些通过命令行都很容易进行求解。

7. 应用领域

在应用界面提供了如下应用领域：

● 结构力学——平面应力问题；

● 结构力学——平面应变问题；

● 静电场问题；

● 静磁场问题；

● 交流电磁场问题；

● 直流导体介质问题；

● 热传导问题；

● 扩散问题。

这些界面都有对话框，它包括 PDE 的系数、边界条件、解的性质等。多数问题不但可用GUI 的方法，而且可用命令行的方法编程解题。

7.2 解偏微分方程的基本方法

7.2.1 一个解偏微分方程的基本例子

解 Poisson 方程：$-\Delta u=f$，边界条件为 Dirichlet 类型。

第一步：激活 MATLAB，键入 PDETOOL，便启动 GUI，然后打开栅格，在

Options 下选择 Grid 和 Snap，调整好栅格间距 Grid Spacing，准备画图（如图 7.2.1-1）。

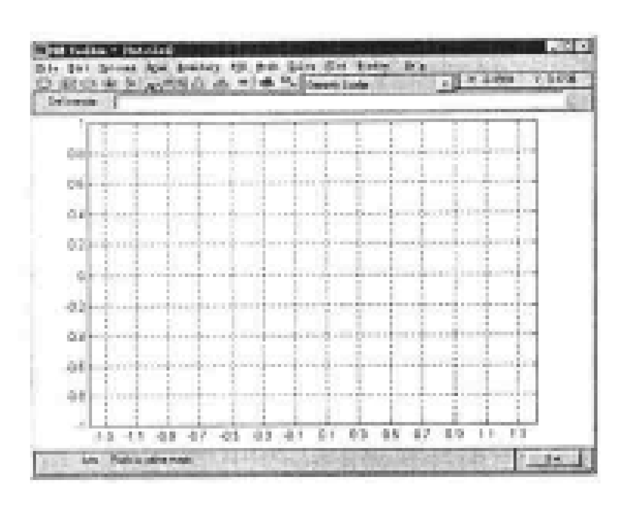


图 7.2.1-1

第二步：分步完成平面几何造型：((SQ1+C1)-C2-SQ2-SQ3+C3+C4)-R1。用菜单或图标，画正方形 SQ1，圆 C1，圆 C2，正方形 SQ2，正方形 SQ3，圆 C3，圆 C4，矩形 R1，然后在 Set formula 栏，用算术运算符将它们连接起来，然后进行布尔运算。若有必要，可以存起来，形成 M 文件（如图7.2.1-2）。

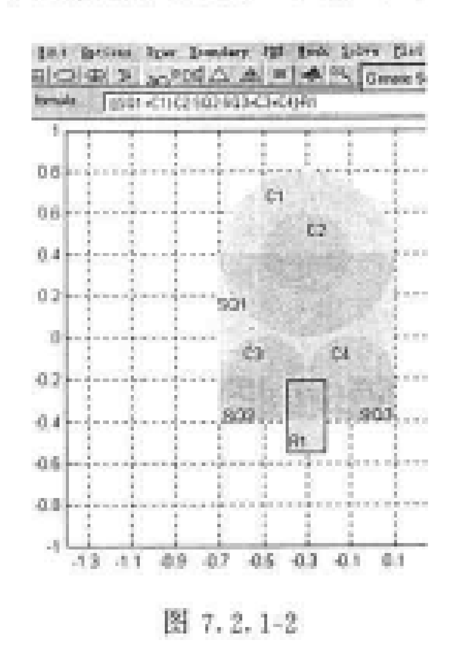


图 7.2.1-2

选择 Boundary 菜单项，进入边界模式 Boundary Mode 后按 ∂Ω，去除子域边界 Remove All Subdomain Borders。如果想要将几何信息和边界信息存起来，应选

择 Export Decomposed Geometry Boundary Cond's 项，然后可将它们分别存于 g，b 变量中，通过 MATLAB 可形成文件。

第三步：选取边界后，选择 Boundary 菜单项，点击 Specify Boundary Condition，在出现的对话框中输入边界条件。本例为缺省条件，即将全部边界设为 Dirichlet 条件，颜色为红色(如图 7.2.1-3 和图 7.2.1-4)。

图 7.2.1-3

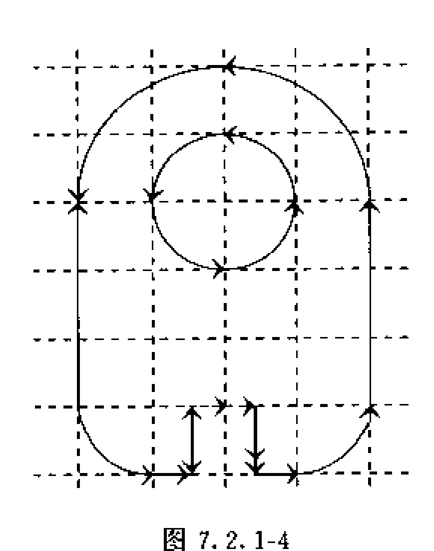


图 7.2.1-4

第四步：选择 PDE 菜单项，在 PDE 模式中，激活 PDE Specification...，设制方程为 $-\Delta(cu)+au=f$ 类型。本例为缺省设置，类型为椭圆型，参数 c，a，f 分别为 1，0，10(如图 7.2.1-5)。

第五步：选择 Mesh 菜单项中的 Initialize Mesh，进行网格剖分(如图

PDE Specification

Equation: -div(c*grad(u))+a*u=f

Type of PDE:

- Elliptic
- Parabolic
- Hyperbolic
- Eigenmodes

Coefficient	Value
c	1.0
a	0.0
f	10.0
d	1.0

OK Cancel

图 7.2.1-5

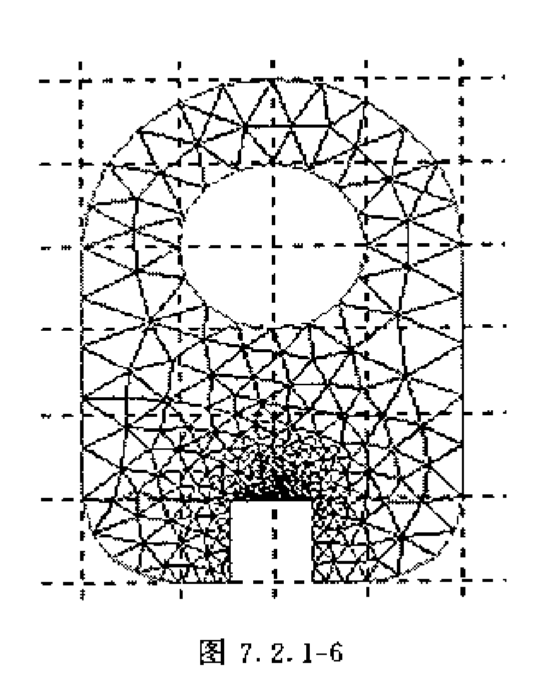


图 7.2.1-6

7.2.1-6)。

第六步：选择 Mesh 菜单项中的 Refine Mesh，进行网格加密（如图 7.2.1-7)。

第七步：选择Solve 菜单项中的Solve PDE，解偏微分方程并显示图形解(如图 7.2.1-8)。

第八步：选择 Plot 菜单项中的 Parameters...，在出现的对话框中，选 Height [3-DPlot]和 Show mesh 项，然后按 Plot 键，可显示三维图形解(如图 7.2.1-9，图 7.2.1-10)。

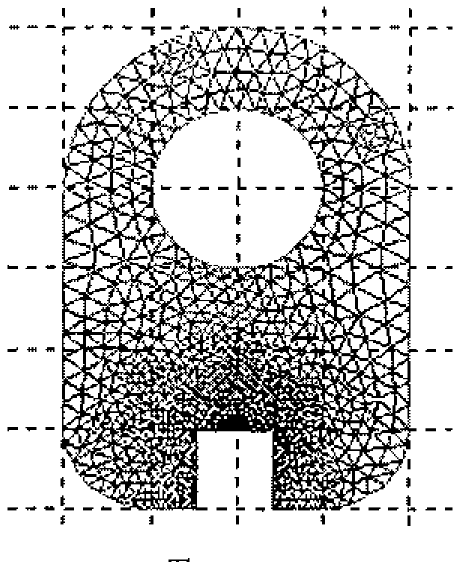


图 7.2.1-7

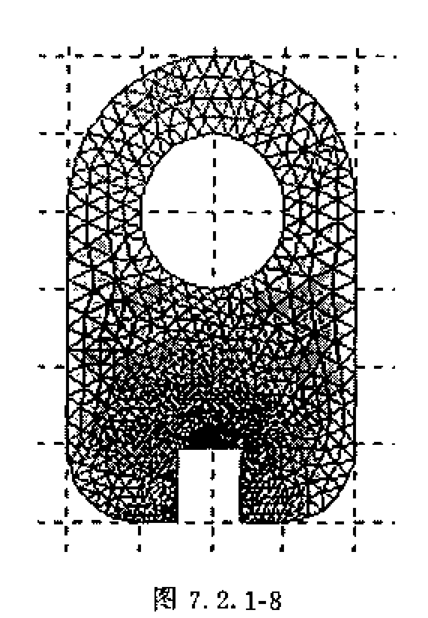


图 7.2.1-8

第九步：如果要画矢量场图，可选择 Plot 菜单项中的 Parameters...，在出现的对话框中，选 Contour 和 Arrows 项，然后按 Plot 键，可显示解的矢量场图(如图 7.2.1-11，图 7.2.1-12)。

图 7.2.1-9

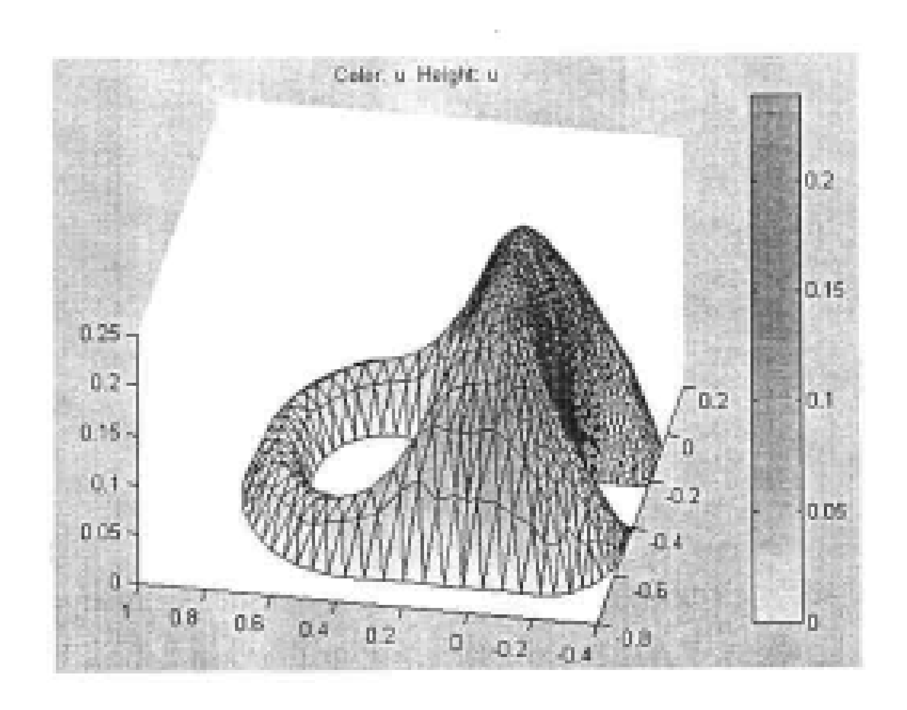


图 7.2.1-10

图 7.2.1-11

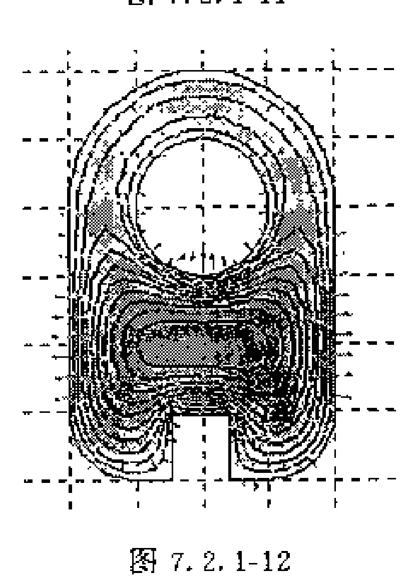


图 7.2.1-12

7.2.2 偏微分方程的图形用户界面

偏微分方程的图形用户界面主要有菜单和工具栏两部分：

1. PDE Toolbox 菜单

(1)**File** 菜单

New	更新或建立一个新的几何结构实体模型 Constructive Solid Geometry(CSG)，并取名为 Untitled。
Open...	从硬盘装载 M 文件。
Save	将在 GUI 内完成的成果储存到一个 M 文件中。
Save As...	将在 GUI 内完成的成果储存到另外一个 M 文件中。

Print...	将PDE工具箱完成的图形送到打印机内进行硬拷贝。
Exit	退出PDE工具图形用户界面。

(2)**Edit** 菜单

Undo	在绘制多边形时退回到上一次操作。
Cut	将已选实体移到剪切板上。
Copy	将已选实体拷贝到剪切板上。
Paste...	将剪切板上的实体拷贝到当前几何结构实体模型中。
Clear	删除已选的实体。
Select All	选择当前几何结构实体造型CSG中的所有的实体及其边界和子域。

(3)**Options** 菜单

Grid	绘图时栅格的开启和关闭。
Grid Spacing...	调整栅格大小。
Snap	捕捉栅格开启和关闭。
Axis Limits...	改变绘图轴的比例。
Axis Equal	绘图轴的打开和关闭。
Turn off Toolbar Help	关闭工具栏按钮的帮助文档。
Zoom	图形缩放的开启和关闭。
Application	应用模式选择。
Refresh	重新显示PDE工具箱中所有的图形实体。

(4)**Draw** 菜单

Draw Mode	进入绘图模式。
Rectangle/Square	以角点方式画矩形/方形(ctrl+鼠标)。
Rectangle/Square (centered)	以中心方式画矩形/方形(ctrl+鼠标)。
Ellipse/Circle	以矩形角点方式画椭圆/圆(ctrl+鼠标)。
Ellipse/Circle (centered)	以中心方式画椭圆/圆(ctrl+鼠标)。
Polygon	画多边形,按右键可封闭多边形。
Rotate...	旋转已选的物体。

Export Geometry Description,Set Formula,Labels...

将几何描述矩阵gd、公式设置字符sf和标识空间矩阵nk输出到主工作空间去。

(5)**Boundary** 菜单

Boundary Mode　进入边界模式。

Specify Boundary Conditions...　对于已选的边界输入条件，如果没有选择边界，则边界条件适用于所有的边界。

Show Edge Labels　显示边界区域标识开关，其数据是分解几何矩阵的列数。

Show Subdomains Labels　显示子区域标识开关，其数字是分解几何矩阵中的子域数值。

Remove Border Subdomain　当图形进行布尔计算时，删除已选取的子域边界。

Remove All Subdomain Borders　当图形进行布尔计算时，删除所有的子域边界。

Export Decomposed Geometry, Boundary Cond's...　将分解几何矩阵 g 和边界条件矩阵 b，输出到主工作空间。

Specify boundary conditions...　定义边界条件。从显示的对话框，你可对已选的边界输入边界条件(如图 7.2.2-1)，共有三种不同的条件类型：

图 7.2.2-1

- 一般 Neumann 条件：这里边界条件是由下面方程系数 q 和 g 确定的，在方程组的情况下，q 是 2×2 矩阵，g 是 2×1 矢量。
- Dirichlet 条件：u 定义在边界上，边界条件方程是 hu=r，这里 h 是可以选择的权因子(通常为 1)。在方程组情况下，h 是 2×2 矩阵，r 是 2×1 矢量。
- 混合边界条件(仅适于方程组情况)：它是 Dirichlet 和 Neumann 的混合边界条件，q 是 2×2 矩阵，g 是 2×1 矢量，h 是 1×2 矢量，r 是一个标量。

(6)**PDE** 菜单

PDE Mode　进入偏微分方程模式。

Show Subdomain Labels　显示子区域标识开关。

PDE Specification...　打开对话框以输入PDE参数和类型(如图7.2.2-2)。

Export PDE Coefficients...　将当前PDE参数c,a,f,d输出到主工作空间，其参数变量为字符类型。

PDE Specification			
Equation: -div(c*grad(u))+a*u=f			
Type of PDE:	Coefficient	Value	
◆ Elliptic	c	1.0	
◇ Parabolic	a	0.0	
◇ Hyperbolic	f	10	
◇ Eigenmodes	d	1.0	
OK		Cancel	

图7.2.2-2

PDE Specification...打开一个对话框，输入偏微分方程类型和应用参数。参数的维数决定于偏微分方程的维数。如果选择专业应用模式，那么特殊偏微分方程和参数将代替标准偏微分方程系数。每一个参数c,a,f和d皆可作为有效的*MATLAB*表达式，以作为计算三角形单元质量中心的参数值。下面的变量是很有用的：

- x和y：x和y坐标。
- u：解。
- ux,uy：解的关于x和y的导数。
- t：时间。

注意：如果偏微分方程参数是解u或者它的导数ux和uy的函数，你必须使用非线性求解器；如果偏微分方程参数是时间t的函数，你必须使用抛物型或双曲型偏微分方程。

(7)**Mesh菜单**

Mesh Mode	输入网格模式。
Initialize Mesh	建立和显示初始化三角形网格。
Refine Mesh	加密当前三角形网格。
Jiggle Mesh	优化网格。
Undo Mesh Change	退回上一次网格操作。
Display Triangle Quality	用0~1之间数字化颜色显示三角形网格的质量，大于0.6的网格是可接受的。

Show Node Labels	显示网格节点标识开关，节点标识数据是点矩阵 P 的列。
Show Triangle Labels	显示三角形网格标识开关，三角形网格标识数据是三角形矩阵 t 的列。
Export Mesh...	输出节点矩阵 P，边界矩阵 e 和三角形矩阵 t 到主工作空间。
Parameters...	打开对话框(如图 7.2.2-3)，修改网格生成参数，网格初始化算法 initmesh 使用网格生成参数有：

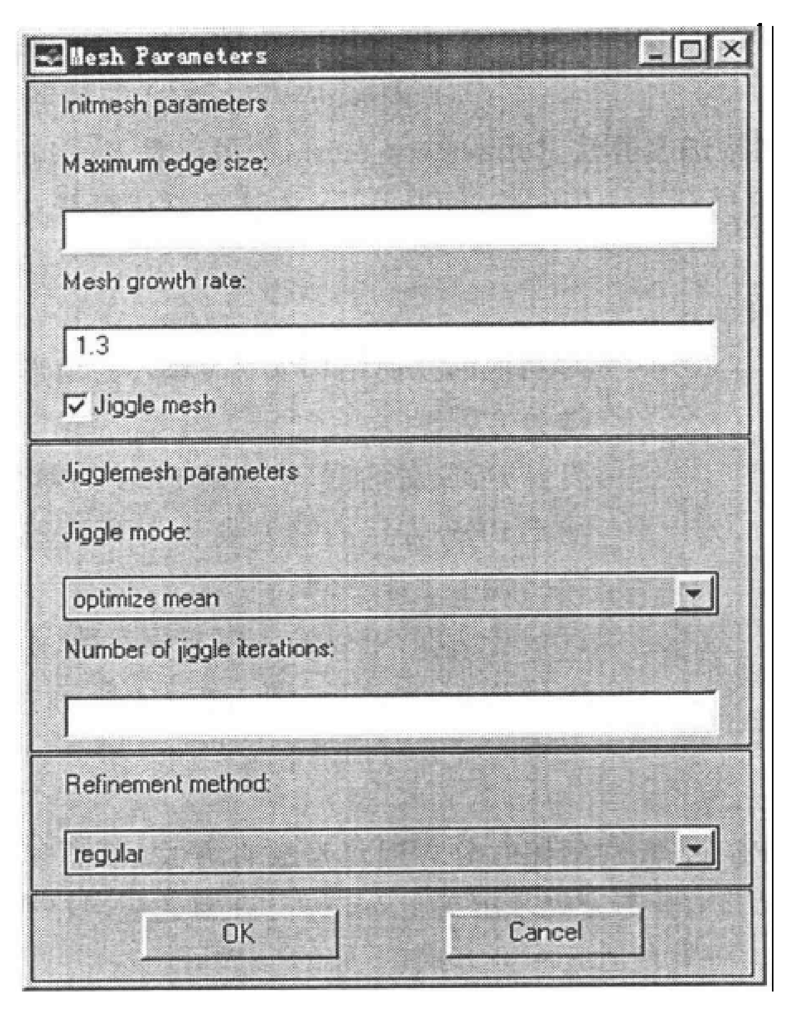


图 7.2.2-3

- Maximum edge size：三角形最大边长度，这个参数是可选的，但必须是实正数。
- Mesh growth rate：网格增长速度。网格的大小随区域的几何尺寸增加而

增加。其数值一定是在1到2之间，缺省值是1.3，即网格增长速率为30%。

- Jiggle mesh：自动优化初始网格开关切换，其算法为jigglemesh。

使用优化网格算法jigglemesh的参数是：

- Jiggle mode：从弹出菜单可选jiggle方式。

可用的方式有：on，optimize minimum和optimize mean。

- On：立即进行生成优化网格。
- Optimize minimum：优化网格反复进行直到达到最小三角形质量或者循环终值。
- Optimize mean：上述选择同样可适用于optimize mean，但它只能增加三角形平均质量。
- Number of jiggle iterations：优化循环次数。对于optimize minimum和optimize mean方式的循环终值，缺省为20。

最后，对于网格加密算法Refinement method可以是regular或longest，缺省方式是regular，它可产生均匀网格；而longest方法常常以最长边优化每一三角形网格。

(8)**Solve** 菜单

Solve PDE	对当前的几何结构实体CSG、三角形网格和图形解偏微分方程。
Parameters...	打开PDE对话框，输入解PDE的参数。
Export Solution...	输出PDE方程的解矢量u。如果可行，将计算特征值1输出到主工作空间。
Parameters...	打开对话框，可以输入解方程参数。每组参数的选择取决于PDE的类型。图7.2.2-4所示为椭圆型偏微分方程解的参数对话框。

- Elliptic PDEs（椭圆型偏微分方程）：为缺省方式，不需要专门定义解方程参数。解椭圆方程采用基本方程求解器assempde。在自适应网格生成和adaptmesh之间可进行选择。对于自适应网格方式，下面的参数可用：

-Adaptive mode：以自适应方式（打开/关闭）切换。

-Maximum number of triangles：三角形网格允许最大数目（可以是无穷大），缺省值则是根据当前三角形网格计算的数值。

-Maximum number of refinements：加密网格最大数目，试图连续加密网格的最大数目。

-Triangle selection method：三角形网格选择方法。PDE工具箱中提供了其中

图 7.2.2-4

两种三角形网格方法，用户也可提供自己的函数。

-Worst triangles：最坏三角形。这个方法是选择比一个最坏三角形的分数值（缺省为 0.5）还要差的所有三角形。详细情况参见 pdeadworst 条。

-Relative tolerance：相对容差。这个方法使用相对容差标准（缺省为 1E－3）的三角形，详细情况参见 pdeadgsc 条。

-User-defined function：用户定义函数。输入用户定义函数，输入用户定义的三角形选择方法。见 pdedemo 的例子。

-Function parameter：函数参数。函数参数允许三角形选择方法微调。对于最坏的三角形方法（pdeadworst），它是用于决定三角形需要调整最坏数的分数值；对于相对容差方法，它是控制更好地适合于 PDE 的容差参数。

-Refinement method：优化方法。有两个选项：regular（均匀的）或 longest（最长边的），可参见“Mesh Menu”菜单项中的 Parameters。如果问题是非线性的，即 PDE 参数是直接依赖于解 u 的，那么必须用非线性解，可采用下面参数：

-Use nonlinear solver：使用非线性解。选择非线性解的切换开关。

-Nonlinear tolerance：非线性容差。对于非线性解的容差参数。

-Initial solution：初始解。一个初始估计值，它可以是当前网格节点上赋以常数 x 和 y 的函数。

例：exp(x. * y)，皆为可选参数，缺省为零。

-Jacobian：Jacobian逼近方法。Fixed固定的(缺省)：不动点迭代、分块的、分块(对角)逼近或者完全Jacobian。

-范数：各种范数是用来计算残差的。取能量范数时输入即为能量，或者对于标量p给出lp范数，缺省是无穷范数。

注意：自适应方式和非线性求解器可一起使用。

● 抛物型偏微分方程(Parabolic PDEs)解抛物型的参数是：

-Time：时间。抛物线型偏微分方程求解时的MATLAB时间矢量。相关时间间隔是依赖于问题的动态状况。

例：0：10 and logspace(-2,0,20)。

-u(t0)：对于抛物型偏微分方程的初始值是u(t0)。初始值可以是一个常数或当前网格的节点值的列向量。

-Relative tolerance：相对容差。对于常微分方程ODE的求解器的相对容差参数，是用来求解抛物型偏微分方程有关时间部分。

-Absolute tolerance：绝对容差。对于常微分方程ODE的求解器的绝对容差参数，是用来求解抛物型偏微分方程有关时间部分。

● Hyperbolic PDEs：双曲型偏微分方程。解双曲型的参数是：

(Time)时间：双曲型偏微分方程求解时的MATLAB时间矢量。相关时间间隔是依赖于问题的动态状况。

例：0：10 and logspace(－2,0,20)。

-u(t0)：对于双曲型偏微分方程的初始值是u(t0)。初始值可以是一个常数或者是当前网格的节点值的列向量。

-u'(t0)：对于双曲型偏微分方程的初始值是u(t0)，可使用与u(t0)相同的格式。

-相对容差：对于常微分方程ODE的求解器的相对容差参数，是用来求解双曲型偏微分方程有关时间部分。

-绝对容差：对于常微分方程ODE的求解器的绝对容差参数，是用来求解双曲型偏微分方程有关时间部分。

图7.2.2-5所示为特征值解的参数对话框。

● Eigenvalue problems：特征值问题。对于特征值偏微分方程，解参数仅仅是特征值求解域，它是一个二元素矢量，在实轴上定义一个区间作为特征值求解域。左边可以是-inf。

例：[0 100]，[-inf 50]。

图7.2.2-6所示为特征值型偏微分方程的解的参数对话框。

(9)**Plot** 菜单

Solve Parameters
Adaptive mode
Maximum number of triangles:
1074
Maximum number of refinements:
10
Triangle selection method:
Worst triangles
Relative tolerance
User-defined function
Worst triangle fraction:
0.5
Refinement method:
longest
Use nonlinear solver
Nonlinear tolerance:
1E-4
Initial solution:
Jacobian:
fixed
Norm:
Inf
OK
Cancel

图 7.2.2-5

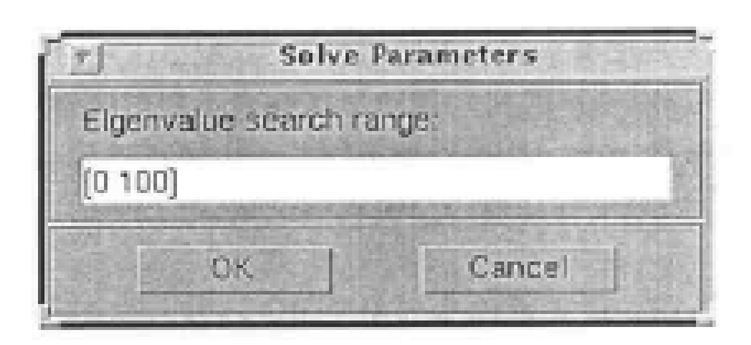


图 7.2.2-6

Plot Solution　　显示图形解。

Parameters...　　打开绘图方式对话框。

Export Movie...　如果动画被录制了，则动画矩阵 M 将输出到主工作空间。

Plot Selection　　对话框(如图 7.2.2-7 所示)。

Parameters...　　打开含有控制绘图和可视化选择的对话框。

对话框上面一行含有四列：

Plot type(位于最左列)：有 6 种不同绘图方式供选择，可用于可视化。

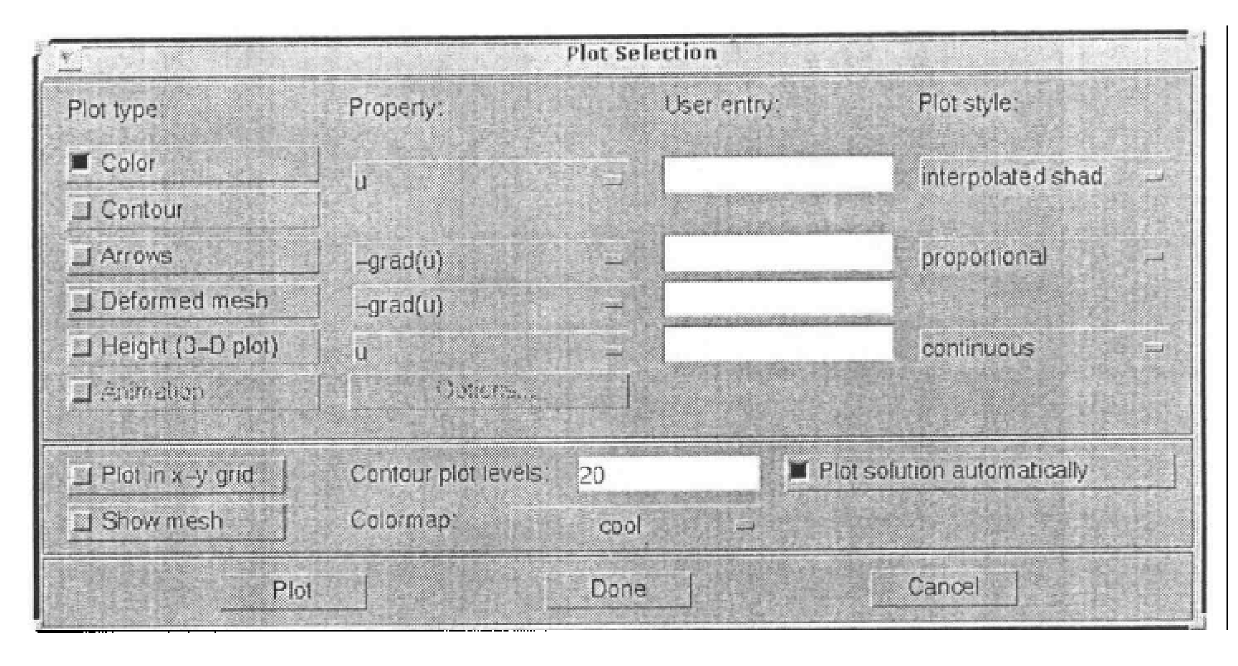


图 7.2.2-7

a) Color(颜色):用于着色曲面标量属性的可视化。

b) Contour(等值线):用于等值线标量属性的可视化。

当绘图类型(颜色和等值线)被检查后,等值线可提高颜色的可视化,等值线被画成黑色。

c) Arrows(箭头):用箭头表示矢量属性的可视化。

d) Deformed mesh(变形网格):用向量属性表示变形网格的可视化。变形会自动地控制在问题区域的10%。这种绘图类型基本上要把结构力学中的x和y位移(u和v)显示,出来。如果没有其他的绘图类型选取,那么变形三角形网格会被显示出来。

e) Height (3-D plot)(三维图形):分别用不同图形窗口进行三维图形(3-D plot)标量属性的可视化。如果颜色和等值线的绘图类型没有选取,则三维图形(3-D plot)绘出的仅仅是网格图,当然也可以在三维图形(3-D plot)中用颜色和/或等值线绘出其他标量属性。

f) Animation(动画):在抛物型和双曲型问题中依赖于时间解的动画。如果选取了这个选项,方程解就被记录下来,然后用move函数在不同的图形窗口中作动画演示。

Property(位于第二列):属性含有4个弹出菜单为不同的属性表,可用来画图时选择相应的绘图类型。

第一个弹出菜单:是用于控制颜色或等值线显示属性。

第二和第三个弹出菜单：使用箭头和变形网格可视化图形表示矢量值属性。

第四个弹出菜单：使用 3-D 中 Z 的高度来控制标量属性。

-u：方程解。

-abs(grad(u))：每个三角形中心的▽u 的绝对值。

-abs(c * grad(u))：每个三角形中心的 c・▽u 的绝对值。

-user entry：MATLAB 表达式，返回一个定义在当前三角形网格的节点或者三角形上的数据矢量。

-solution u：u 的导数是 ux 和 uy。c・▽u 的分量 cux 和 cuy，以及 x 和 y 都是局部工作空间的变量。你可以将表达式输入到弹出菜单的右边的 User entry 的编辑框里面。

例：u. * u，x+y。

矢量属性弹出菜单设有下面属性：

-grad(u)：u 的负梯度-▽u。

-c * grad(u)：c 乘以 u 的负梯度-c・▽u。

—user entry：MATLAB 表达式[px；py]可返回一个 $2\times n$ 维定义当前三角形网格数据矩阵。解 u，其导数 ux 和 uy，c・▽u 的 x 和 y 分量，cux 和 cuy，以及 x 和 y，皆适用于局部空间。三角形中心的值是由节点插值得到的。你可以在右边 User entry 中对属性弹出式菜单进行赋值。

例：[ux；uy]，[x；y]。

对于方程组情形，若使用颜色、等值线或三维绘图等可视化属性是 u，v，abs(u，v)和用户输入框。

若使用箭头或变形网格，你可选择(u，v)或用户框。对于结构力学中的应用来说，u 和 v 分别是 x 和 y 方向的位移。

User entry：含有 4 个编辑框，可供用户输入自己的表达式。如果用户在编辑窗左边的弹出菜单选择了 user entry ，便可输入属性；否则相应的编辑框是暗淡的。

绘图方式含有 3 个弹出菜单，分别可用作对颜色、箭头及三维绘图的属性控制。用于绘制彩色曲面的绘图属性 plot styles 是：

-Interpolated shading：插值着色。使用已选的色图 colormap 和插值着色，即对于每个三角形区域用线性插值进行着色(缺省)。

-Flat shading：使用已选的色图和平坦方式着色，即对每个三角形区域进行单色着色。

对于箭头绘制有两种方式可选择：

-Proportional：箭头的长度与你设置的有关属性大小相对应。

-Normalized：所有的箭头的长度都是相等的，这对于只想了解矢量场的方向是很有用的，即使区域很小，矢量的方向也会清晰可见。

对于三维绘图 height (3-D plots)，绘图方式是：

-Continuous（连续的）：从三角形中点到网格节点用插值方法产生一个光顺的连续的曲面。

-Discontinuous（离散的）：产生一离散曲面，但每一三角形区域其数据和高度为常数。

解总共为三个属性（两个标量属性和一个矢量场）可同时显示。如果三维绘图 Height (3-D plot)的选择被关掉，那么在 pdetool GUI 的主轴上画出二维图形解；否则会在不同的图形窗口内，绘出三维图形。

辅助绘图控制选择：在对话框底部有许多辅助绘图控制选择项。

● Plot in x-y grid：如果选择了此项，图形解将原来的三角形网格转变成矩形 x-y 网格。这对于动画来说是非常有用的，因为四边形网格可有效地加速动画片存储过程。

● Show mesh：在曲面图中，如果选择此项，网格被画成黑色；如果缺省，网格消隐。

● Contour plot levels：等值线条数。例如，可输入 15 或 20，然后系统按此数目画出等值线，缺省为 20。

例：[0:100:1000]，logspace(−1,1,30)

● Colormap：使用 Colormap 弹出菜单，可以选择不同的色图：cool，gray，bone，pink，copper，hot，jet，hsv 和 prism。

● Plot solution automatically：如果关闭此项，则 PDE Toolbox 不会立刻显示图解。然而，新解仍然可以用这个对话框画出来。

对于抛物型和双曲型 PDE，对话框的右边底部含有辅助选择（如图 7.2.2-8）：

Animation：如果动画是可用的，在其属性域中，可以看到一个 Options 按钮，把它按下去，一个附加对话框出现，其中含有动画方面的控制参数。

Animation rate (fps)：对于动画参数，可控制每秒显示画面数目 (fps)。

● Number of repeats：动画播放循环次数。

Replay movie：如果选择了此项，当前的动画片将重新播放；如果当前内存没有动画片，这个选择是不能用的。

对于特征值问题，右边底部设有关于所有的特征值的弹出式菜单。图形解是关于已选的特征值所对应的特征矢量。缺省时最小特征值将被选择。

对于三维图形，均处于 Rotate 3d on 状态，用鼠标可动态显示图形。

Plot：如果按下此按钮，按当前图形设置，其解立刻被绘出；如果当前没有偏微分方程，则首先解方程，有解以后，再绘图。

Done：如果选择了此按钮，对话框则会关闭。当前的设置将被储存，没有新图产生。

Cancel：对话框关闭，设置也不会改变。

图 7.2.2-8

(10)Window 菜单

从 Window 菜单项，可选择当前打开的所有的 MATLAB 图形窗口，选择后的窗口移至前台。

(11)Help 菜单

Help...　　显示简洁帮助窗口。

About...　　显示带有一些程序信息的窗口。

2. PDE 工具栏



主菜单下工具栏含有许多图标按钮，可以快速、简洁地运行 PDE 函数和菜单项。按钮的功能从左到右按序排列：左边五个按钮为绘图模式，其余 6 个为边界、网格、解方程和图形显示控制功能，最右边的为图形缩放功能键：

：以角点方式画矩形/方形(ctrl+ 鼠标)。

：以中心方式画矩形/方形(ctrl+ 鼠标)。

：以矩形角点长轴方式画椭圆/圆(ctrl+ 鼠标)。

：以中心方式画椭圆/圆(ctrl+ 鼠标)。

：画多边形，按右键可封闭多边形。

∂Ω：进入边界模式。

PDE：打开 PDE 对话框。

：初始化三角形网格。

：加密三角形网格。

：解偏微分方程。

：解的三维图形，打开 Solution Plot Selection 对话框。

：显示缩放切换按钮。

7.3 解偏微分方程的举例

7.3.1 椭圆型方程

【例 7.3.1-1】 设一长直接地金属槽，如图 7.3.1-1 所示，其侧壁与底面电位为 0，顶盖电位为 $100\sin\frac{\pi}{a}x$，求槽内电位分布。

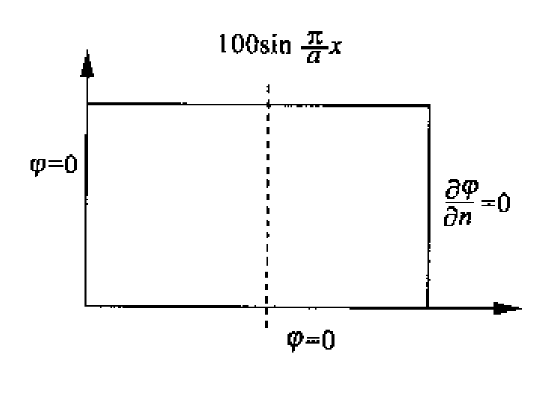


图 7.3.1-1

解：此题可化为二维场问题。电位函数 φ 满足拉普拉斯方程，构成混合型边值问题。

$$\begin{cases}\dfrac{\partial^2\varphi}{\partial x^2}+\dfrac{\partial^2\varphi}{\partial y^2}=0\\ \varphi|_{(x=0,0\leqslant y\leqslant 0.5a)}=\varphi|_{(0\leqslant x\leqslant 0.5a,y=0)}=0\\ \varphi|_{(0\leqslant x\leqslant 0.5a y=0.5a)}=100\sin(\dfrac{\pi}{a}-x)\\ \dfrac{\partial\varphi}{\partial n}|_{(x=0.5a,0\leqslant y\leqslant 0.5a)}=0\end{cases}$$

解题步骤：

(一)区域设置：选择菜单 Draw 下的 Rectangle/Square，画矩形。

(二)输入边界条件：进入 Boundary Mode 或按 ∂Ω，输入：

1. 左边界：狄利克莱 Dirichlet 条件：h=1，r=0。
2. 右边界：诺依曼 Neumann 条件：g=0，q=0。
3. 上边界：狄利克莱 Dirichlet 条件：h=1，r=100 * sin(pi * x)。
4. 下边界：狄利克莱 Dirichlet 条件：h=1，r=0。

(三)方程参数设定：点击 PDE，在 PDE Spacification 对话框内选择 Elliptic：c=1；a=0；f=0。

(四)网格剖分：点击△按钮或选择菜单 Initialize Mesh。

(五)图形解显示参数设置：点击菜单 Plot 下的 Parameter，在对话框中选择 Contour 和 Arrows 两项。

(六)解方程：点击 = 按钮或选择 Solve PDE 菜单，结果如图 7.3.1-2 所示。

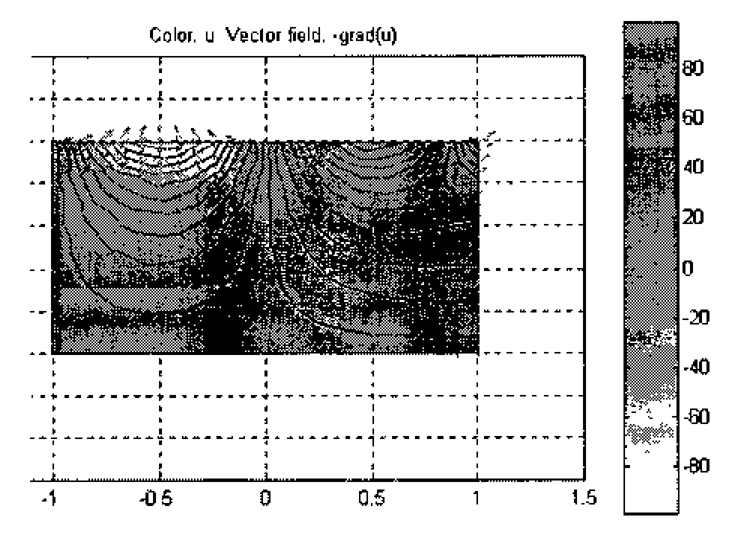


图 7.3.1-2

【例 7.3.1-2】 图 7.3.1-3 为一个同轴电缆的示意图，两个同芯长方形导体之间充满了线性介质。假设导体之间加有直流电压 10V，内导体接地，导体之间密度为 ρ，传输线的长度充分长，则理想的认为电场在传输线各个截面上的分布相同，因此顺序求出电场在截面上的分布，即求出电场在各导体间的分布，从而能检查绝缘材料的工作状况。

由于同轴电缆的几何形状、介质及激励都关于 x 轴和 y 轴对称，只需求解整个截面四分之一即可，如图 7.3.1-4。

边值条件：

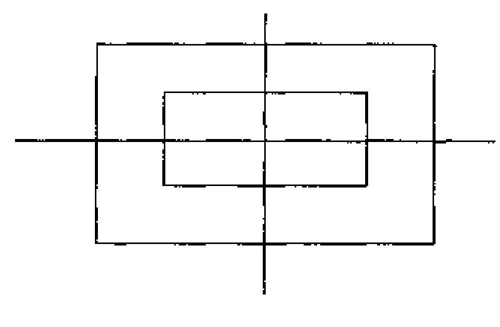


图 7.3.1-3

$$\nabla^2\varphi=-q \quad x,y\in\Omega$$

$$q=\frac{\rho}{\varepsilon}$$

狄利克莱 Dirichlet 条件：

$$\varphi|_{r1}=0$$

$$\varphi|_{r3}=10$$

诺依曼 Neumann 条件：

$$\frac{\partial\varphi}{\partial n}\Big|_{r2}=\frac{\partial\varphi}{\partial n}\Big|_{r4}=0$$

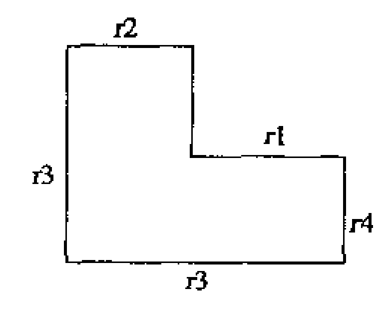


图 7.3.1-4

解题步骤：

（一）区域设置：选择菜单 Draw/Polygon，画 L 多边形。

（二）输入边界条件：按 后，输入

1. 左边界 r3：狄利克莱 Dirichlet 条件：h=1，r=10。
2. 右边界 r4：诺依曼 Neumann 条件：g=0，q=0。
3. 上边界 r1：狄利克莱 Dirichlet 条件：h=1，r=0。
4. 上边界 r2：诺依曼 Neumann 条件：g=0，q=0。
5. 下边界 r3：狄利克莱 Dirichlet 条件：h=1，r=10。

（三）方程参数输入：先点击 按钮，进入 PDE Mode，再选 PDE Spacification，在对话框中选择 Elliptic，输入：c=1；a=0；f=$-\rho/\varepsilon$=10。

（四）网格剖分：按 或 Initialize Mesh。

（五）图形解显示参数设置：点击菜单 Plot 下的 Parameter，在对话框中选择 Contour 和 Arrows 两项。

（六）解方程：点击按钮 或菜单 Solve PDE，结果如图 7.3.1-5 所示。

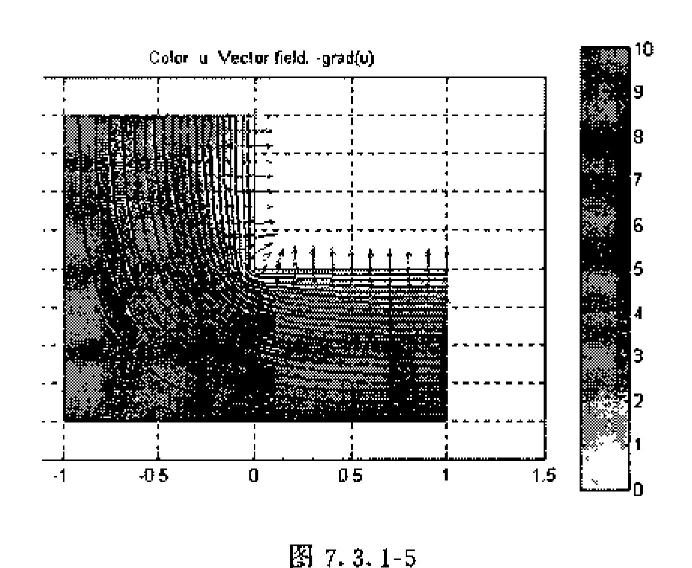


图 7.3.1-5

【例 7.3.1-3】 设有理想流体在图 7.3.1-6 所示的域内自左向右流动。流速势 $\varphi(V=-grad\varphi)$ 满足拉普拉斯方程，试计算流场的离散网格点上的 φ 值，并绘出等势线和流线。混合型的边界条件如图 7.3.1-6 所示。

解题步骤：

一、有限元法

（一）区域设置：选择菜单 Draw 下的 Rectangle/Square，画大矩形，其角点：(1,1)(21,11)，小矩形角点：(16,5)(21,7)，右边靠齐。

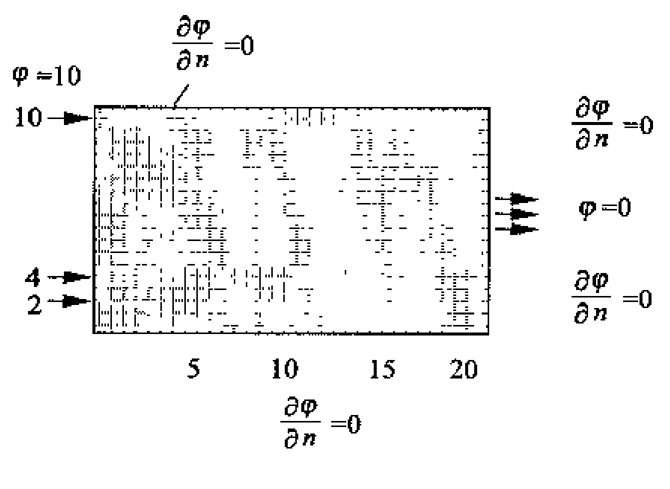


图 7.3.1-6

(二)输入边界条件：按[∂Ω]后，输入：

1. 左边界：狄利克莱 Dirichlet 条件：h=1，r=10。
2. 右上边界：诺依曼 Neumann 条件：g=0，q=0。
3. 右中边界：狄利克莱 Dirichlet 条件：h=1，r=0。
4. 右下边界：诺依曼 Neumann 条件：g=0，q=0。
5. 上边界：诺依曼 Neumann 条件：g=0，q=0。
6. 下边界：诺依曼 Neumann 条件：g=0，q=0。

(三)方程定义：先点击[PDE]按钮，再选 PDE Spacification，在对话框中选择：Elliptic，输入：c=1；a=0；f=0。

(四)网格剖分：按[△]或菜单 Initialize Mesh。

(五)图形解显示参数设置：点击菜单 Plot 下的 Parameter，在对话框中选择 Contour 和 Arrows 两项。

(六)解方程：点击按钮[=]或菜单 Solve PDE，结果如图 7.3.1-7 所示。

二、若用插分方法，程序如下：

```
FI=10;hx=21;hy=11;
v1=ones(hy,hx);
     %左右 Dirichlet 条件边界值：
       v1(:,1)=ones(hy,1) * FI;v1(5:7,hx)=zeros(3,1);
v2=v1 ; maxt=1;t=0;      %初始化
while (maxt>0.0000001)      %由 v1 迭代，算出 v2，迭代精度为 0.0000001
%   for k=1:1500
   for i=2:hy-1
```

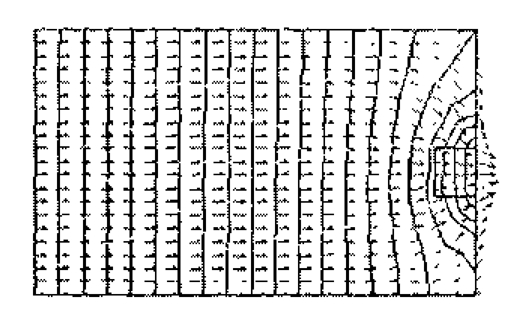


图 7.3.1-7

```
    for j=2:hx-1
v2(i,j)=(v1(i,j-1)+v1(i,j+1)+v1(i-1,j)+v1(i+1,j))/4;%用五点格式差分
    t=abs(v2(i,j)-v1(i,j));
      maxt=0;
        if(t>maxt) maxt=t; end
          end
    end
v2(1,2:hx-1)=v2(2,2:hx-1);
v2(hy,2:hx-1)=v2(hy-1,2:hx-1);
v2(2:4,hx)=v2(2:4,hx-1);v2(8:10,hx)=v2(8:10,hx-1);
  v1=v2;
end
contour(v2);    %画等值线
hold on;    %保屏
    x=1:1:21;y=1:1:11;
[xx,yy]=meshgrid(x,y);    %形成栅格
[Gx,Gy]=gradient(v2,0.6,0.6);    %计算梯度
quiver(xx,yy,Gx,Gy,'r'); axis([0,25,0,11])    %画矢量图
hold off
```

【例 7.3.1-4】 变压器结构如图 7.3.1-7 所示。假定变压器铁芯为具有相对磁导率为 2 000 的线性材料，另一种为相对磁导率为 1 的空气。线圈绕组所占的面积流过密度为 10^{+5}A/m^2 的电流。为简化计算，假定变压器铁芯外沿的磁势为零，这就成为狄利克莱条件的一部分。

变压器结构及激励关于 x 轴对称，因此沿 x 轴有齐次的诺伊曼条件成立；变压器的几何结构关于 y 轴对称，且激励关于 y 轴反对称，因此沿 y 轴存在着磁势为零的狄利克莱条件。这样只需选取变压器的四分之一。

变压器可划分为14个单元、13个节点，其中单元(1)到单元(8)及单元(12)、单元(13)为铁芯，单元(9)到单元(11)及单元(14)为空气材料，单元(9)到单元(11)含有电流激励($J=-10^{+7}\mathrm{A/m^2}$)。这一静磁场的描述方程为泊松方程及拉普拉斯方程。由于只考虑二维结构，磁势只在沿 z 轴的方向不为零，而沿其他方向的分量都为零。对于这样一个二维问题的矢量磁势 **A**，实际上简化为一个标量 A_z($\mu_0=4\pi\times10^{-7}$，μ_r 为不同介质系数)。

$\mu_r=2\,000$　(∈单元1～8，12，13)

$\mu_r=1$　(∈单元9～11，14)

$\varphi=0$　(∈节点1～7)

$\frac{\partial\varphi}{\partial n}=0$　(∈节点8～10)

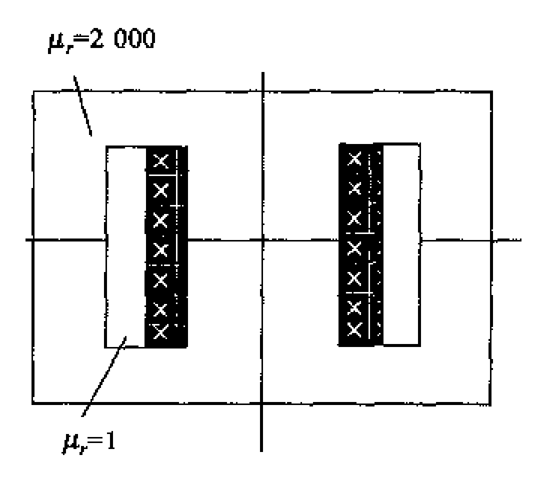


图 7.3.1-7

(一)区域设置：选择菜单 Draw 下的 Rectangle/Square，画大矩形、小矩形，下边靠齐。

(二)输入边界条件：按∂Ω后，输入：

1. 左边界：狄利克莱 Dirichlet 条件：h=1，r=0。
2. 右边界：狄利克莱 Dirichlet 条件：h=1，r=0。
3. 上边界：狄利克莱 Dirichlet 条件：h=1，r=0。
4. 下中边界：诺依曼 Neumann 条件：g=0，q=0。
5. 下左边界：狄利克莱 Dirichlet 条件：h=1，r=0。
6. 下右边界：狄利克莱 Dirichlet 条件：h=1，r=0。

(三)方程参数设置：单击PDE后，选择菜单 PDE Spacification，在对话框中输

人方程参数。

在大矩形选择 Elliptic：c=1；a=0；f=4 * pi * 2000；

在小矩形选择 Elliptic：c=1；a=0；f=4 * pi。

（四）网格剖分：按[图标]或 Initialize Mesh。

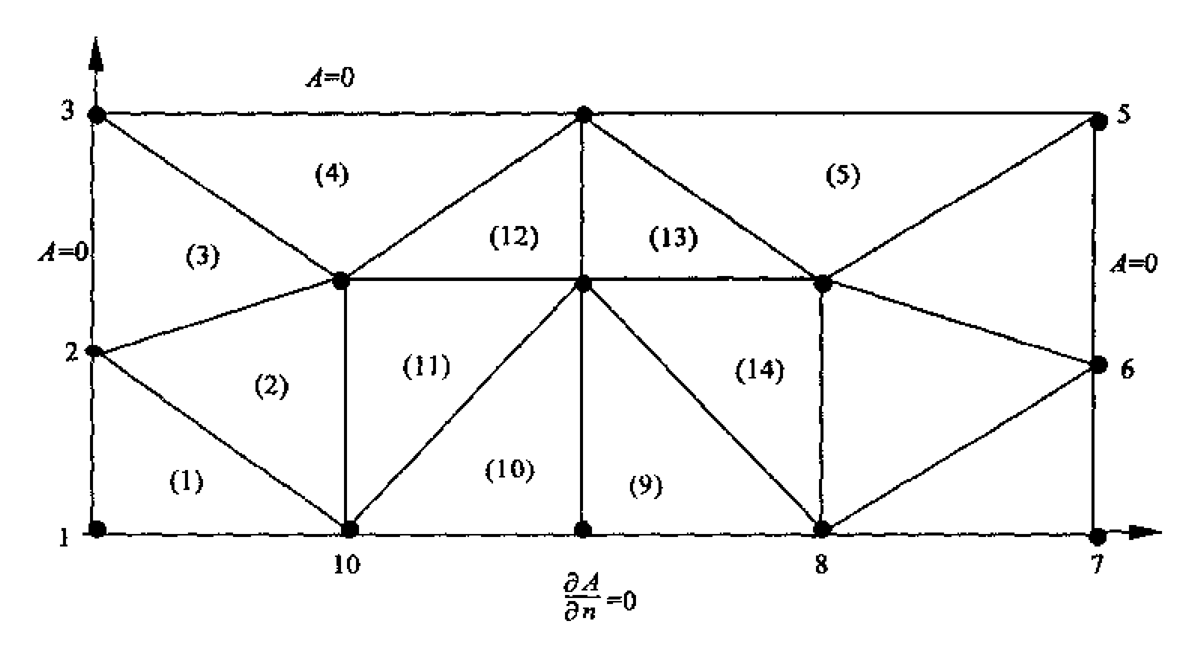


图 7.3.1-8

（五）图形解显示参数设置：点击菜单 Plot 下的 Parameter，在对话框中选择 Contour 和 Arrows 两项。

（六）点击按钮[图标]或菜单 Solve PDE，结果如图 7.3.1-9 所示。

【例 7.3.1-5】 在导电模拟的实验研究中，制备了如图 7.3.1-10 所示的二维电场模型，其内两种导电媒质的电导率分别为 γ_1 和 γ_2（$\gamma_1=2\gamma_2$），它们在场域的对角线 L 上接合。电极间外施电压 10V。求由此两种媒质构成的二维电流场内的电位分布。

解：由于存在两种不同的导电媒质，所以应分别定义相应的电位函数 φ_1 和 φ_2，它们都满足拉普拉斯方程，构成混合型边值条件：

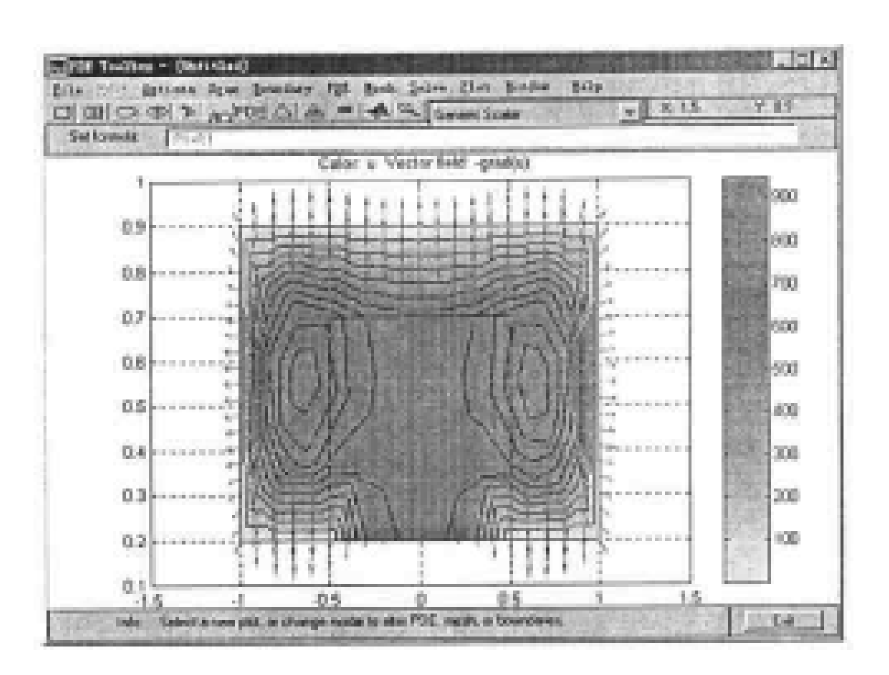


图 7.3.1-9

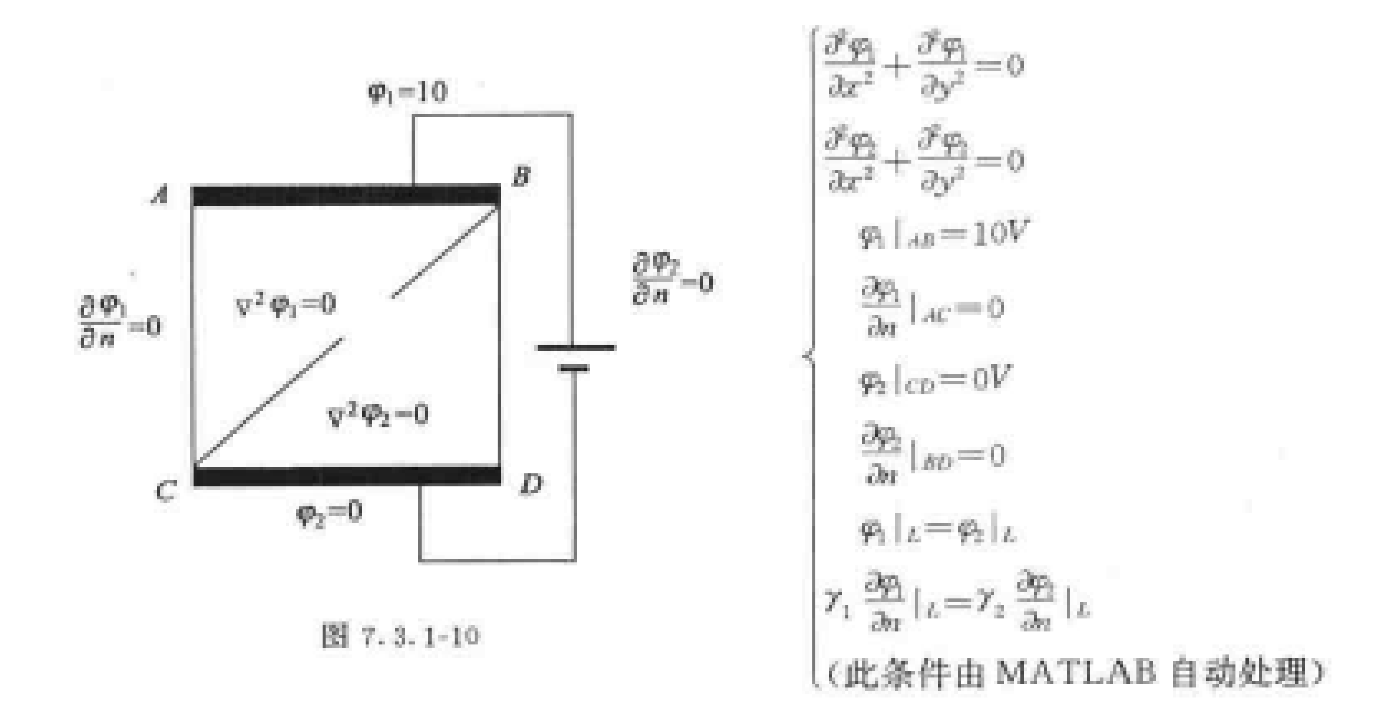


图 7.3.1-10

$$\begin{cases}\dfrac{\partial^2\varphi_1}{\partial x^2}+\dfrac{\partial^2\varphi_1}{\partial y^2}=0\\ \dfrac{\partial^2\varphi_2}{\partial x^2}+\dfrac{\partial^2\varphi_2}{\partial y^2}=0\\ \varphi_1|_{AB}=10V\\ \dfrac{\partial\varphi_1}{\partial n}|_{AC}=0\\ \varphi_2|_{CD}=0V\\ \dfrac{\partial\varphi_2}{\partial n}|_{BD}=0\\ \varphi_1|_L=\varphi_2|_L\\ \gamma_1\dfrac{\partial\varphi_1}{\partial n}|_L=\gamma_2\dfrac{\partial\varphi_2}{\partial n}|_L\\ \text{(此条件由 MATLAB 自动处理)}\end{cases}$$

(一)区域设置：选择 grid，Axis Equal，snap 后，画上三角形、下三角形。

(二)输入边界条件：按 $\partial\Omega$ 后，在出现的对话框中输入：

1. 左边界：诺依曼 Neumann 条件：g=0，q=0。

2. 上边界：狄利克莱 Dirichlet 条件：h=1，r=10。

3. 下边界：狄利克莱 Dirichlet 条件：h=1，r=0。

4. 右边界：诺依曼 Neumann 条件：g=0，q=0。

(三)方程参数设置：单击PDE后，设定拉普拉斯方程。

在上三角形选择 Elliptic：c=1；a=0；f=0。

在下三角形选择 Elliptic：c=1；a=0；f=0。

(四)网格剖分：按△或 Initialize Mesh。

(五)图形解显示参数设置：点击菜单 Plot 下的 Parameter，在对话框中选择 Contour 和 Arrows 两项。

(六)点击按钮＝或菜单 Solve PDE，结果如图 7.3.1-11 所示。

【例 7.3.1-6】 横截面为矩形的无限长槽由 3 块接地导体板构成，槽的盖板接直流电压 V0，求矩形槽中的电位分布。如图 7.3.1-12，

$$\Delta u=0,\Omega=\{0<x<17,0<y<10\}$$

$$u(0,y)=u(17,y)=u(x,0)=0$$

$$u(x,10)=100$$

一、使用 UGI 方法

解题步骤：

(一)区域设置：矩形

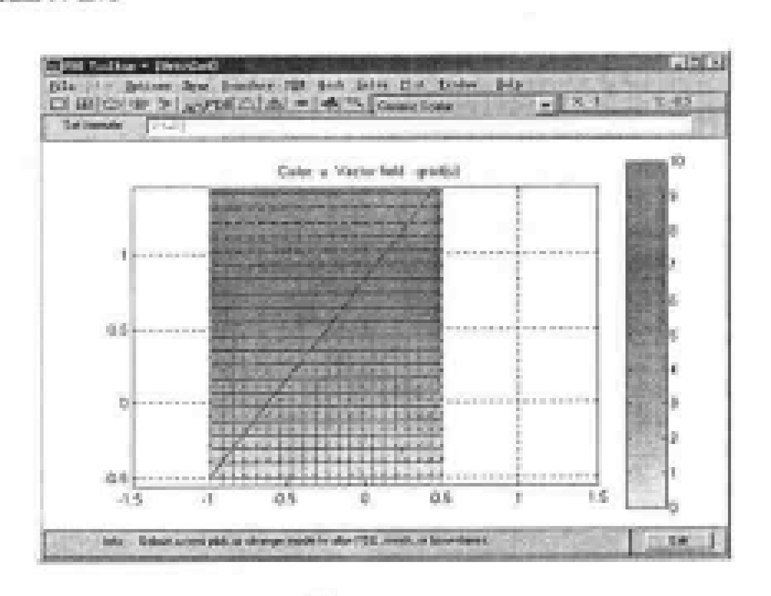


图 7.3.1-11

(二)输入边界条件：进入边界模式或按∂Ω，输入：

1. 左边界：狄利克莱 Dirichlet 条件：h=1，r=0。

2. 右边界：狄利克莱 Dirichlet 条件：h=1，r=0。

3. 上边界：狄利克莱 Dirichlet 条件：h=1，r=100。

4. 下边界：狄利克莱 Dirichlet 条件：h=1，r=0。

（三）方程参数设定：点击PDE按钮或菜单 PDE Spacification，在出现的对话框中选择 Elliptic：c＝1；a＝0；f＝0。

（四）网格剖分：按△或菜单 Initialize Mesh。

（五）解方程：按=按钮或 Solve PDE 菜单，结果如图 7.3.1-13 所示。

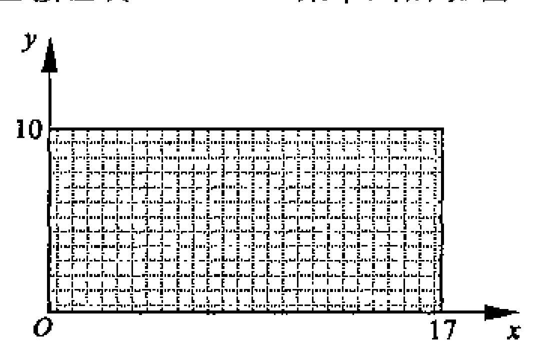


图 7.3.1-12

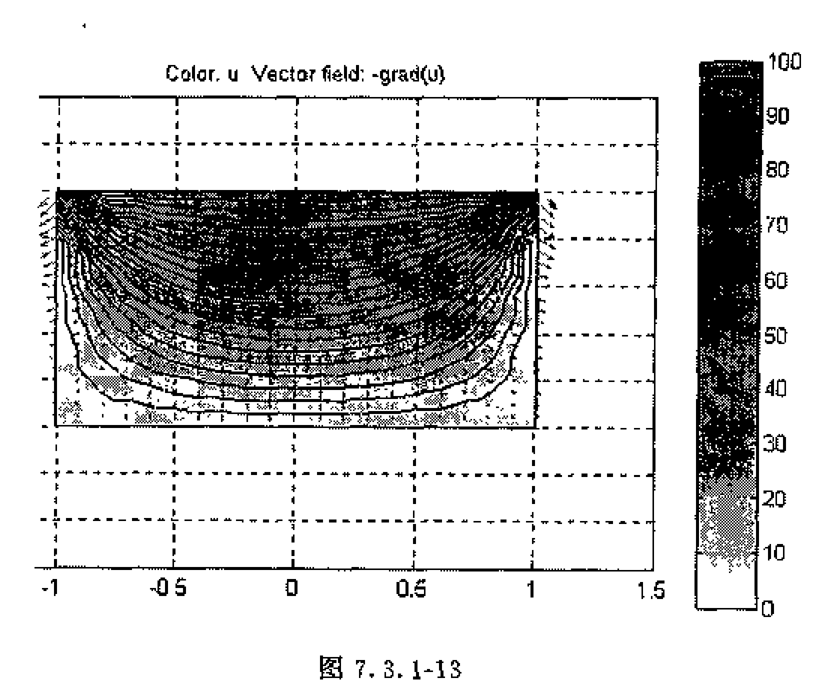


图 7.3.1-13

二、差分方法程序如下：

```
v0=100;hx=17;hy=10;
v1=zeros(hy,hx);
v1(hy,:)=ones(1,hx)*v0;   %上边界值
v1(2:hy-1,2:hx-1)=ones(hy-2,hx-2);   %赋初值
v2=zeros(hy,hx); maxt=1;t=0;   %初始化
```

```
v2(hy,:)= v1(hy,:);   %上边界值
while (maxt>0.1)  %由v1迭代,算出v2,迭代精度为0.1
  for i=2:hy-1
    for j=2:hx-1
    %用五点格式差分
    v2(i,j)=(v1(i,j-1)+v1(i,j+1)+v1(i-1,j)+v1(i+1,j))/4;
  t=v2(i,j)-v1(i,j);
    maxt=0;
      if(t>maxt) maxt=t; end
        end
      end
  v1=v2;
end
subplot(1,2,1),surf(v2) %如图7.3.1-14所示,左图为曲面图
axis([0,17,0,10,0,100])
    subplot(1,2,2),
contour(v2);%在右图画等值线
hold on;  %保屏
x=1:1:17;y=1:1:10;
```

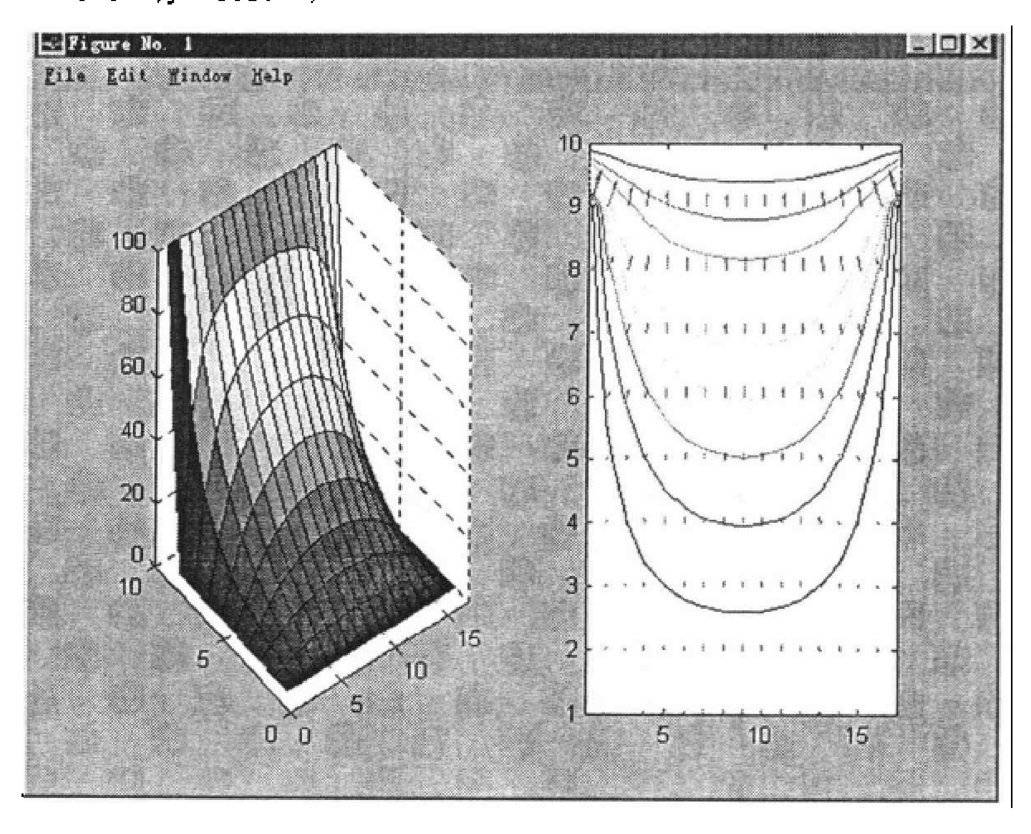


图 7.3.1-14

```
[xx,yy]=meshgrid(x,y);  %形成栅格
[Gx,Gy]=gradient(v2,0.6,0.6);  %计算梯度
quiver(xx,yy,Gx,Gy,'r'),  %画矢量图
hold off
```

7.3.2 抛物型方程

热传导方程,金属板的导热问题

【例 7.3.2-1】 考虑一个带有矩形圆孔的金属板上的热传导问题。板的中间保持 100℃,板的外侧四周热量从板内向外空气定常流动,其他边及内孔边界保持绝缘。初始 $t=t_0$ 时板的温度为 0℃,于是概括为如下定解问题:

$$\begin{cases} d\dfrac{\partial u}{\partial t}-\Delta u=0 & \\ u=100 & \text{内圆四条边界狄利克莱 Dirichlet 条件:h}=1,r=0; \\ \dfrac{\partial u}{\partial n}=-10 & \text{外面四条边界诺依曼 Neumann 条件:g}=0,q=-10; \\ \dfrac{\partial u}{\partial n}=0 & \text{其他边界上.} \\ u|_{t=0}=0 & \end{cases}$$

使用 *GUI* 求解这一问题:

(一)区域设置:单击菜单 *Rectangle/Square*,在窗口拖拉出一个矩形,双击矩形区域,显示矩形区域 R1;同样办法作内圆孔 C1。点击菜单 Ellipse/Circle (Centered),然后在 Set formula 内键入 R1－C1。

(二)边界条件设置:单击∂Ω,使边界变红色,然后分别双击每条边界,出现对话框设置边界条件。

(三)方程设置:单击PDE,在 PDE Specification 对话框中设置 parabolic:d = 1, c = 1, a = 0, f = 0,单击 OK。

(四)网格剖分:单击△,或者加密网格单击△。

(五)初值和误差的设置:单击 Solve,再单击 Solve parameters,键入 time=0:5, u(t0)=0, Relative tolerance=0.01, Absolute tolerance=0.001,单击 OK。

(六)数值解的输出:单击 Solve,再单击 Export Solution,在对话框输入 u,单击 OK。再在 MATLAB 命令窗口键入 u,按回车键,这时显示按节点编号的数值解。

(七)解的图形:单击☰,窗口显示 time = 5 时解的彩色图形。如图 7.3.2-1 所示。

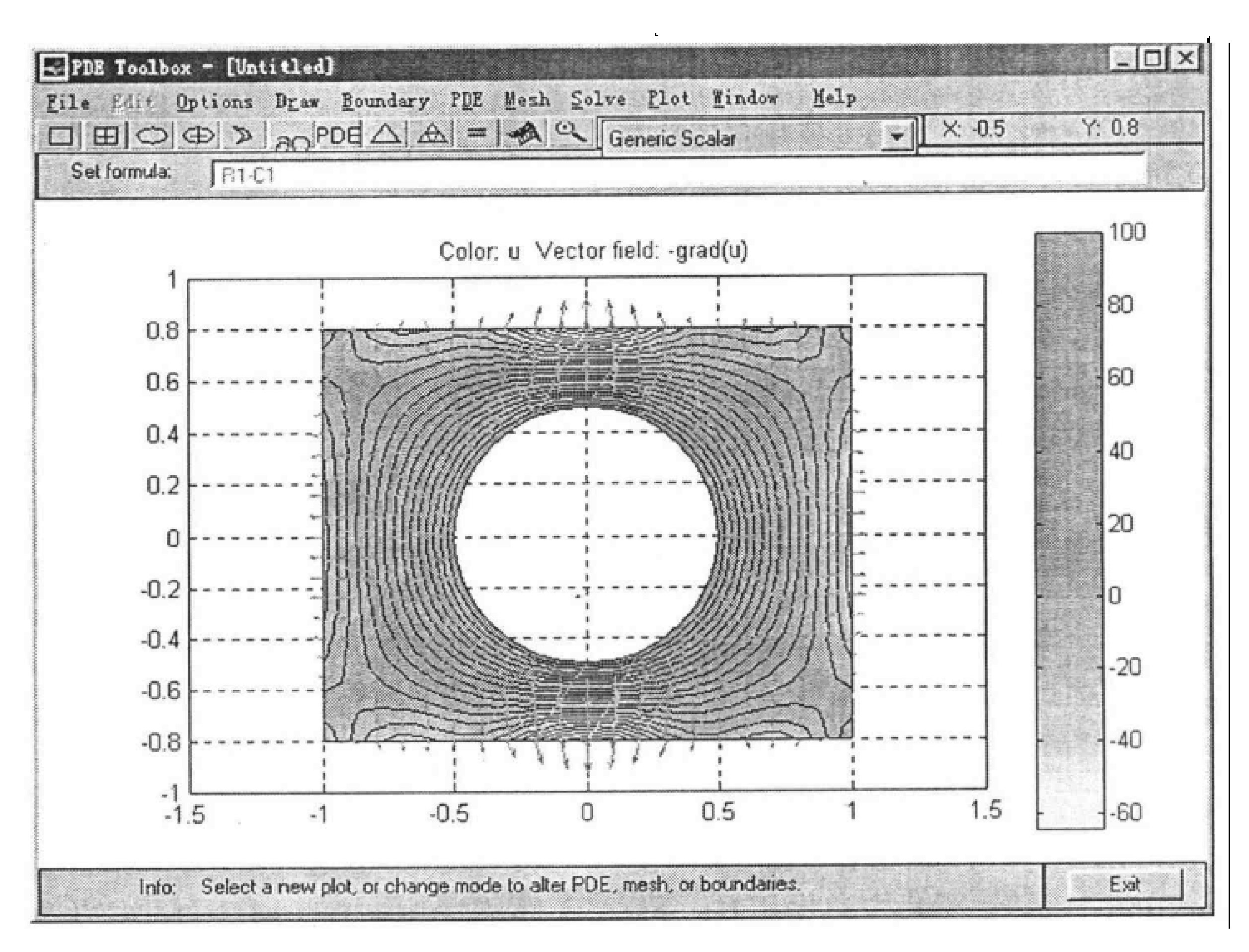


图 7.3.2-1

放射性杆的热扩散

这是一个三维的抛物型方程问题，现在利用柱坐标将它简化成二维问题。考虑一个圆柱形放射性杆，其左端供热，右端保持常温，侧面与环境有热交换。由于放射性作用，热量均匀地产生，初始温度为 0℃。于是可以用如下方程描述：

$$\rho c\frac{\partial u}{\partial t}-\nabla\cdot(k\nabla u)=f$$

其中 ρ 为密度，c 为杆的热容量，k 为导热系数，f 为放射性热源密度。

【例7.3.2-2】一金属杆的密度 ρ 取为 7 800 kg/m^3，热容量 c 为 500 ws/kg℃，导热系数 k 为 40 w/m℃，热源密度 f 为 20 000 w/m^3，右端恒温为 100℃，侧面环境温度为 100℃，热交换系数为 50 w/m^2℃，左端的热流为 5 000 w/m^2。

现在取柱坐标 (r,θ,z)，因为关于轴对称，故与 θ 无关，从而化为仅与 (r,z) 有关的二维方程：

$$r\rho c\frac{\partial u}{\partial t}-\frac{\partial}{\partial r}\left(kr\frac{\partial u}{\partial r}\right)-\frac{\partial}{\partial z}\left(kr\frac{\partial u}{\partial z}\right)=fr$$

边界条件：

在杆的左端（$z=-1.5$ 处）：

$\boldsymbol{n}\cdot(k\nabla u)=5\,000$，由于 Toolbox 中 $\boldsymbol{n}\cdot(c\nabla u)+qu=g$，$c$ 与 r 有关 $c=kr$，因此边界条件写成 $\boldsymbol{n}\cdot(c\nabla u)=5\,000r$。

在杆的右端（$z=1.5$ 处）：$u=100$。

在杆的侧面（$r=0.2$ 处）：$\boldsymbol{n}\cdot(k\nabla u)=50(100-u)$，在 Toolbox 中写成

$\boldsymbol{n}\cdot(c\nabla u)+50r\cdot u=50r\cdot 100$。

在杆的对称轴（$r=0$ 处）：属于非边界，需要给出人为强制边界条件

$\boldsymbol{n}\cdot(c\nabla u)=0$，初始温度：$u|_{t=t_0}=0$。

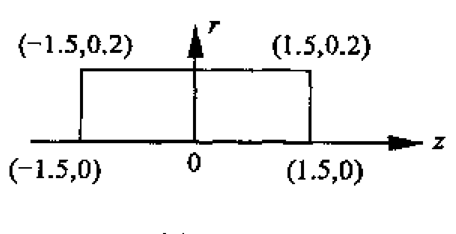


图 7.3.2-2

下面利用 GUI 解这一问题。将杆的对称轴 z 方向记为 x 轴，杆的半径 r 方向记为 y 轴。

（一）区域设置：作矩形，设置端点坐标为（−1.5，0），（1.5，0.2）。形成长为 3，半径为 0.2 的杆的半截面区域。如图 7.3.2-2 所示。

（二）边界条件设置：单击 ∂Ω，使边界变红。双击边界，打开 Boundary Condition 对话框。然后左边界用 Neumann 条件，键入 q＝0，g＝5 000y；右边界用 Dirichlet 条件，键入 h＝1，r ＝100；侧面边界用 Neumann 条件，键入 q＝50y，g＝50y100；轴心用 Neumann 条件，键入 q＝0，g＝0。

（三）方程设置：单击 PDE，在 PDE Specification 对话框中设置 parabolic：c＝40 * y，a＝0，d＝7 800 500y，f＝20 000y。

（四）解的参数设置：单击 Solve，再单击Parameters...，设置 $u(t_0)=0$，*Relative tolerance*＝0.01，*Absolute tolerance*＝0.001，单击 *OK*。

（五）网格剖分：点击 △ 按钮。

（六）解的图形显示：还可以设置动画，按每 1000 秒显示一次，计算到 20 000 秒。我们可以看到热量的流动，如图 7.3.2-3 所示。为此只需要单击 Plot，再单击 Parameters...，在 Plot Selection 对话框中，选 Animation。

如果在 PDE Specification 对话框中设置 Elliptic，这时得到定常问题的解，也就是抛物型方程在 $t\to+\infty$ 的解。

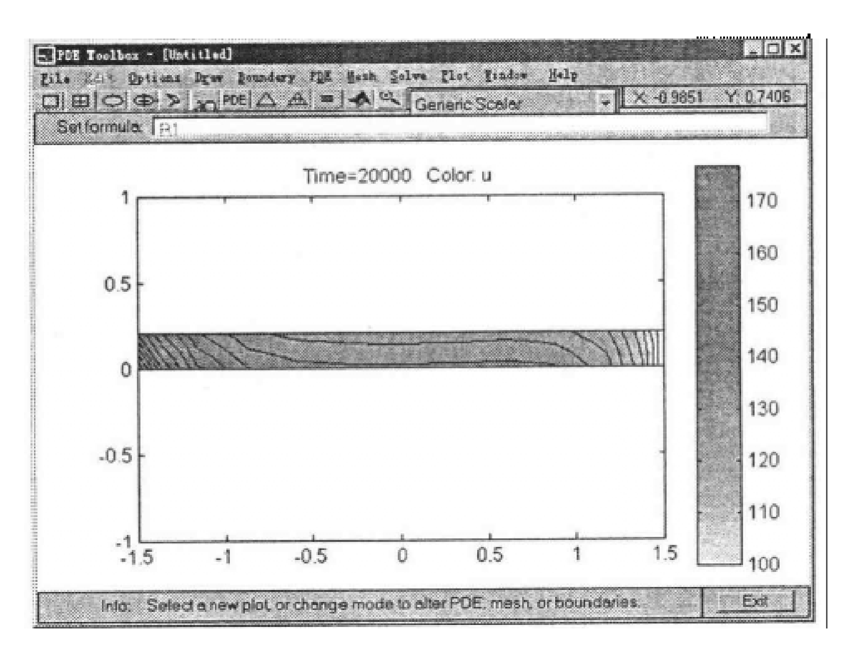


图 7.3.2-3

7.3.3 双曲型方程

【例 7.3.3-1】 考虑如下二维波动方程定解问题：

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2}-\Delta u=0 \qquad (x,y)\in\Omega=\{(x,y)\mid -1<x<1,-1<y<1\}, \\ u|_{x=\pm1}=0 \\ \dfrac{\partial u}{\partial n}|_{y=\pm1}=0 \\ \quad t=0 \\ u=\arctan\left(\cos\left(\dfrac{\pi}{2}x\right)\right) \\ \dfrac{\partial u}{\partial t}=3\sin(\pi x)e^{\sin\left(\frac{\pi}{2}y\right)} \end{cases}$$

用GUI求解：

（一）作正方形区域。

（二）设置边界条件。

（三）作网格剖分。

（四）方程设置：打开 PDE Specification 对话框，设置 hyperbolic，键入 c＝1，a＝0，f＝0，d ＝1。

（五）参数设置：在 Solve 菜单中打开 Solve Parameters 对话框，在 time 栏中键入 Linspace(0,5,31)，以及 u 的初始值 u(t0)＝atan(cos(pi/2 * x))，u(t0)＝3 *

sin(pi * x)*exp(sin(pi/2 * y))。

(六)解方程:点击按钮。结果如图7.3.3-1所示。

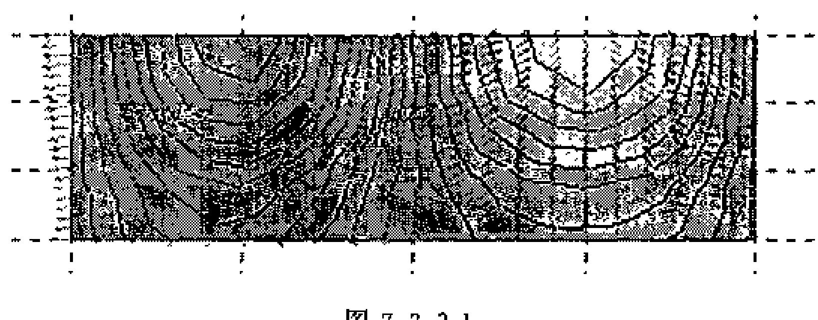


图 7.3.3-1

7.3.4 特征值问题

【例7.3.4-1】 先考虑一个方域上的特征值问题:

$$\begin{cases} -\Delta u=\lambda u \quad (x,y)\in\Omega=\{(x,y)\mid 0<x,y<1\} \\ u\mid_{\partial\Omega}=0 \end{cases}$$

这个问题有精确解:特征值为 $\lambda_{mn}=\pi^2(m^2+n^2)$, $m,n=1,2,3,\ldots$,对应的特征函数为 $u_{mn}=\sin \pi mx \cdot \sin \pi ny$.

现在用GUI求解,使用Generic Scalar。

(一)先作方域,四个端点坐标为(0,0),(1,0),(1,1),(0,1)。设置边界条件u=0。下面定义特征值问题:单击PDE,在PDE Specification对话框中选Eigenmodes,PDE的系数设置为c=1, a=0, d=1,单击OK。再设置特征值范围,如考虑求小于100的特征值,只要单击Solve,再单击Parameters...,键入[0 100],单击OK。

(二)网格剖分:单击,再单击加密,设置Plot中的图形要求。

(三)求解:单击,这时显示的解为对应第一特征值Lambda(1)= 19.928 1的特征函数 $u_{1,1}$ 的图形。如果要输出小于100的特征值及对应的特征函数,只要单击Solve,再单击Export Solution,键入U I,单击OK。再在MATLAB命令窗口输入U,按回车键,立即显示按节点编号的特征函数值。如果输入I,按回车键,立即显示特征值:19.928 1, 50.468 6, 50.499 7, 82.028 7。它们分别为 $\pi^2(1^2+1^2)$, $\pi^2(1^2+2^2)$, $\pi^2(2^2+1^2)$, $\pi^2(2^2+2^2)$ 的近似值。如果要作第2,第3…特征值对应的特征函数图形,只要在Plot Selection选定某一特征值,单击OK,立即可以得到关于这个特征值对应的特征函数的图形。如图7.3.4-1 (a)(b)(c)所示。

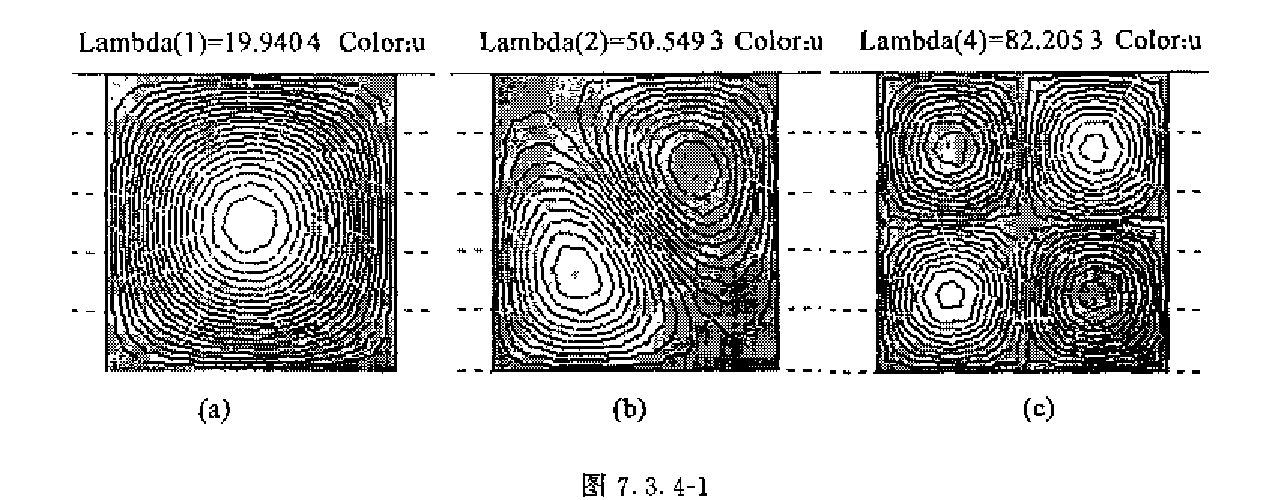


图 7.3.4-1

7.3.5 应用模型

一、应力与应变

【例 7.3.5-1】 一块带孔钢板，下方两边被钳住，其上方的圆弧断面上受拉力，其他边均自由。假设钢板厚为 1mm，钳住的边皆为 0.12m，上方圆的圆心为(0.6,0.6)，半径为 0.2。杨氏模量为 196×10^3 (MN/m^2)，Poisson 比为 0.31。圆弧边界受外法向荷载为 500N/m，考虑钢板厚，所以表面牵引力为 $0.5MN/m^2$。

我们感兴趣的是计算 x 和 y 方向的应变和应力、剪切应力以及 Mises 等效应力。

使用 GUI 求解。

(一)打开 PDE Toolbox GUI，选择 Application mode，选择 Structural Mechanics, Plane Stress。

(二)区域设置：在 Options 下，选择 Axes Limits，键入 x:[0 1.0]，y:[0 1.0]，单击 Apply。然后选择 Grid Spacing...，选择非 AUTO 方式，键入 x:[0:0.2:1.0]，y:[0:0.2:1.0]。最后点击 Axes Equal 和 Snap。

输入图形：单击菜单 Draw 下的 Rectangal/Square，键入角点坐标(0.4, 0.6)，(0.8, 0.2)，得到正方形 SQ1。再选 Ellipse/Circle(centered)，画大圆 C1：圆心(0.6,0.6)，半径 0.2。画小圆 C2：圆心(0.6,0.6)，半径 0.1。画矩形 R1，角点坐标(0.5,0.5)，(0.8,0.1)。最后在公式栏 Set formula 中，键入((SQ1+C1)−C2)−R1。

(三)边界条件设置：单击 Boundary，下边的直角边是固定的，即满足齐次

Dirichlet 条件，上方两端圆弧边界处满足 Neumann 条件：q＝0，g1＝0.5＊nx，g2＝0.5＊ny，其余边界是自由的，即满足齐次 Neumann 条件：q＝0，g＝0。

（四）方程设置：打开 PDE Specification 对话框，键入参数：杨氏模量 E＝196E3，Poisson 比 v＝0.31，因为没有受力，故 K_x 和 K_y 为 0，而 ρ(rho)在本例中不使用。如图 7.3.5-1 所示。

PDE Specification

Equation: Structural mechanics, plane stress

Type of PDE: Elliptic / Parabolic / Hyperbolic / Eigenmodes

Coefficient	Value	Description
E	196E3	Young's modulus
nu	0.31	Poisson ratio
Kx	0.0	Volume force, x-direction
Ky	0.0	Volume force, y-direction
rho	1.0	Density

OK Cancel

图 7.3.5-1

（五）网格剖分：单击，再单击加密。

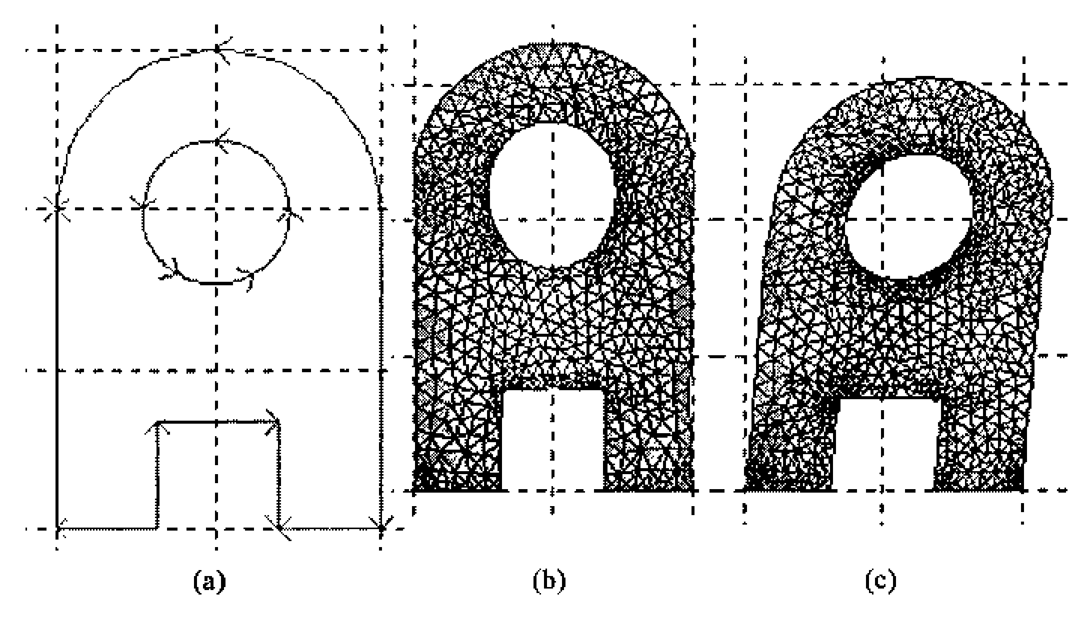
（六）求解：单击 Plot，再单击 Plot Selection，在菜单 Plot Paramter 下生成的对话框 Plot Selection 中设置输出各种变量的图形，如 u 和 v 方向位移，x 和 y 方向应变、应力、剪切应力、剪切应变、von Mises 等效应力以及主应力、应变等等。利用变形网格可以作出 von Mises 有效应力和位移图形，如图 7.3.5-2(b)所示，如果仅右上圆弧受力，有效应力和位移图形如图 7.3.5-2(c)所示。此法也可以用来分析应力集中等情况。

二、静电场问题

这一应用模型包括高压设备、电子仪器和电容器等实际问题。所谓“静”是指场对于时间的变化很缓慢，波长要比所考虑的区域尺度大的多。在静电学中，电位 V 与场强 $\boldsymbol{E}$ 的关系为 $\boldsymbol{E}=-\nabla V$，考虑在各向同性介质中 $\boldsymbol{D}=\varepsilon\boldsymbol{E}$，由 Maxwell 方程知，电位移矢量 $\boldsymbol{D}$ 满足 $\nabla\cdot\boldsymbol{D}=\rho$，于是得到关于电位 V 的 Poisson 方程

$$-\nabla\cdot(\varepsilon\nabla V)=\rho$$

其中 ε 是介电系数，ρ 是空间电荷密度。这里 ε 实际上可写成 $\varepsilon\varepsilon_0$，ε_0 是真空介



(a)　(b)　(c)

图 7.3.5-2

电系数(8.854×10^{-12}法拉第/米)，ε 是相对介电系数。不同介电材料介电系数是不同的，例如空气为 1.000 59，变压器油为 2.24 等等。

【例 7.3.5-2】 考虑一个正方形构件的静电势问题，它的外边界边长为 1.6，圆孔直径为 0.8，在内边界上电势保持为 1 000V，而在外边界电势保持为 0V，它们均为 Dirichlet 条件。在区域内不含电荷，于是导出 Laplace 方程 $\Delta V=0$。

下面使用 GUI 求解：

(一)选择应用模型中的 Electrostatics。

(二)区域设置：作中心在原点、边长为 0.8 的正方形 SQ1，再作中心在原点、半径为 0.4 的圆 C1。在 Set formula 键入 SQ1－C1。

(三)定义边界条件：双击内边界键入 Dirichlet 条件为 1 000，外边界也设定 Dirichlet 条件为 0。再打开 PDE Specification 对话框，设置电荷密度(rho)为 0，介电系数(epsilon)为 1(由于 Laplace 方程是齐次的，所以介电系数值不影响结果)。

(四)网格剖分：单击△。

(五)解出方程：单击=。

(六)如果使用 Adaptive mode，可以在角点这种梯度大的区域加密网格，以提高解的精度。点击菜单 Solve/Parameters，在出现的对话框中设置最大三角形数达到 500。然后单击 Refine，使得网格均匀加密。最后再单击=，便得到解的图形。如果要作等势线图形，只要在 Plot selection 对话框选 contour。若想显示每 50V 一

条等势线，从 0 到 1 000 共 20 条等势线，选择 Color per leavers 为 20 。如图 7.3.5-3所示。

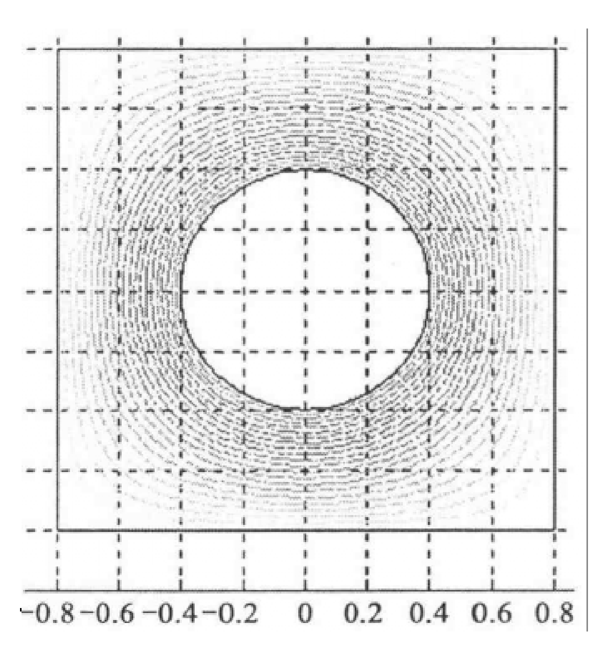


图 7.3.5-3

三、静磁场问题

在磁铁、电动机、变压器这样一类问题中都有静磁场问题。所谓"静"是指关于时间的变化缓慢，因而从如下定常的 Maxwell 方程出发：

$$\nabla \times \boldsymbol{H}=\boldsymbol{J}$$

$$\nabla \cdot \boldsymbol{B}=0$$

$$\boldsymbol{B}=\mu \boldsymbol{H}$$

其中 $\boldsymbol{B}$ 是磁感强度，$\boldsymbol{H}$ 是磁场强度，$\boldsymbol{J}$ 是电流密度，μ 是材料的磁导率。由于 $\nabla \cdot \boldsymbol{B}=0$，故存在一个静磁矢势 $\boldsymbol{A}$，使得

$$\boldsymbol{B}=\nabla \times \boldsymbol{A}$$

及

$$\nabla \times (\frac{1}{\mu} \nabla \times \boldsymbol{A})=\boldsymbol{J}$$

平面问题中假设电流平行于 z 轴，故 $\boldsymbol{A}$ 仅有 z 分量

$$\boldsymbol{A}=(0,0,A),\boldsymbol{J}=(0,0,J)$$

从而，上述方程可以简化成椭圆型 PDE

$$-\nabla \cdot (\frac{1}{\mu} \nabla \boldsymbol{A})=\boldsymbol{J},$$

其中 $\boldsymbol{J}=J(x,y)$.

对于二维情况，可以计算磁感应强度为

$$\boldsymbol{B}=\left(\frac{\partial A}{\partial y},\frac{\partial A}{\partial x},0\right)$$

磁场强度为

$$\boldsymbol{H}=\frac{1}{\mu}\boldsymbol{B}$$

对于不同性质的材料组成的区域之间的交界面上，$\boldsymbol{H}\times\boldsymbol{n}$ 是连续的，也就是 $\frac{1}{\mu}\frac{\partial A}{\partial n}$ 连续。由于使用有限元方法解 PDE 问题，从变分问题出发，故交界面上这一条件作为自然边界条件处理。

在强磁性材料中，μ 是依赖于场 $|\boldsymbol{B}|=|\nabla \boldsymbol{A}|$，从而导致非线性解。

如果是 Dirichlet 边界条件，则要求在边界上给出磁势 $\boldsymbol{A}$ 的值。如果是 Neumann 边界条件，则要求在边界上给出 $\boldsymbol{n}\cdot(\frac{1}{\mu}\nabla A)$ 的值，也就是在边界上给出磁场强度 $\boldsymbol{H}$ 的切向分量。磁势 $\boldsymbol{A}$，磁场强度 $\boldsymbol{H}$ 以及磁感强度 $\boldsymbol{B}$ 的可视化均可实现，还可以作 $\boldsymbol{B}$ 和 $\boldsymbol{h}$ 的向量场图形。

【例 7.3.5-3】 考虑由双极电动机定子线圈产生的静磁场问题。假设马达很长，使得端点的影响可以忽略不计，从而简化成二维模型处理。

区域由 4 部分组成：

- 两个铁磁体：定子和转子；
- 定子和转子之间的间隙；
- 带有直流的转子线圈。

在空气和线圈中磁导率 μ 为 1，而定子和转子的磁导率为

$$\mu=\frac{\mu_{max}}{1+c\parallel\nabla(\boldsymbol{A})\parallel^2}+\mu_{min}$$

其中 $\mu_{max}=5000$，$\mu_{min}=200$，$c=0.05$，这些是由变压器钢材决定的。

- 线圈的电流密度 $\boldsymbol{J}$ 为 1，其他均为 0。

这一问题的磁势 $\boldsymbol{A}$ 关于 y 轴是对称的，关于 x 轴是反对称的，从而只需考虑 $x\geqslant0$ 和 $y\geqslant0$ 部分，而且在 x 轴上满足 Neumann 条件 $\boldsymbol{n}\cdot(\frac{1}{\mu}\nabla\boldsymbol{A})=0$，在 y 轴上满足 Dirichlet 条件 $\boldsymbol{A}=0$。由于马达外面的磁场可以忽略，故外边界均为 Dirichlet 条件 $\boldsymbol{A}=0$。

使用 GUI 求解：

这一问题比较复杂，它的区域包括 5 个圆和 2 个矩形，它的方程在不同区域也不完全相同，而且包括非线性方程。下面详细介绍利用 GUI 求解的过程。

1. 首先在应用模式中选定Magnetostatics，并且打开Solve选Parameters，再打开Solve Parameters对话框，选定Use nonlinear solver。

2. 区域设置：在Options打开Grid Spacing对话框，键入x：(-1.5 1.5)，y：(-1 1)，网格步长为0.1。键入如下命令函数：

```
pdecirc(0,0,1,)
pdecirc(0,0,0.8,)
pdecirc(0,0,0.6,)
pdecirc(0,0,0.5,)
pdecirc(0,0,0.4,)
pderect([-0.2 0.2 0.2 0.9],)
pderect([-0.1 0.1 0.2 0.9],)
pderect([0 1 0 1],)
```

然后在Set formula键入(C1+C2+C3+C4+C5+R1+R2)*SQ1，所得图形如图7.3.5-4所示。然后要删去图中的部分线段和弧段，具体做法如下：先单击Boundary Mode，再单击所要删去的线段或弧段，单击Boundary中的Remove Subdomain Border，这样就删去了所要删的线段和弧段。最后使得整个区域由4部分组成：转子区域1，定子区域2，空隙区域3，线圈区域4，如图7.3.5-5所示。

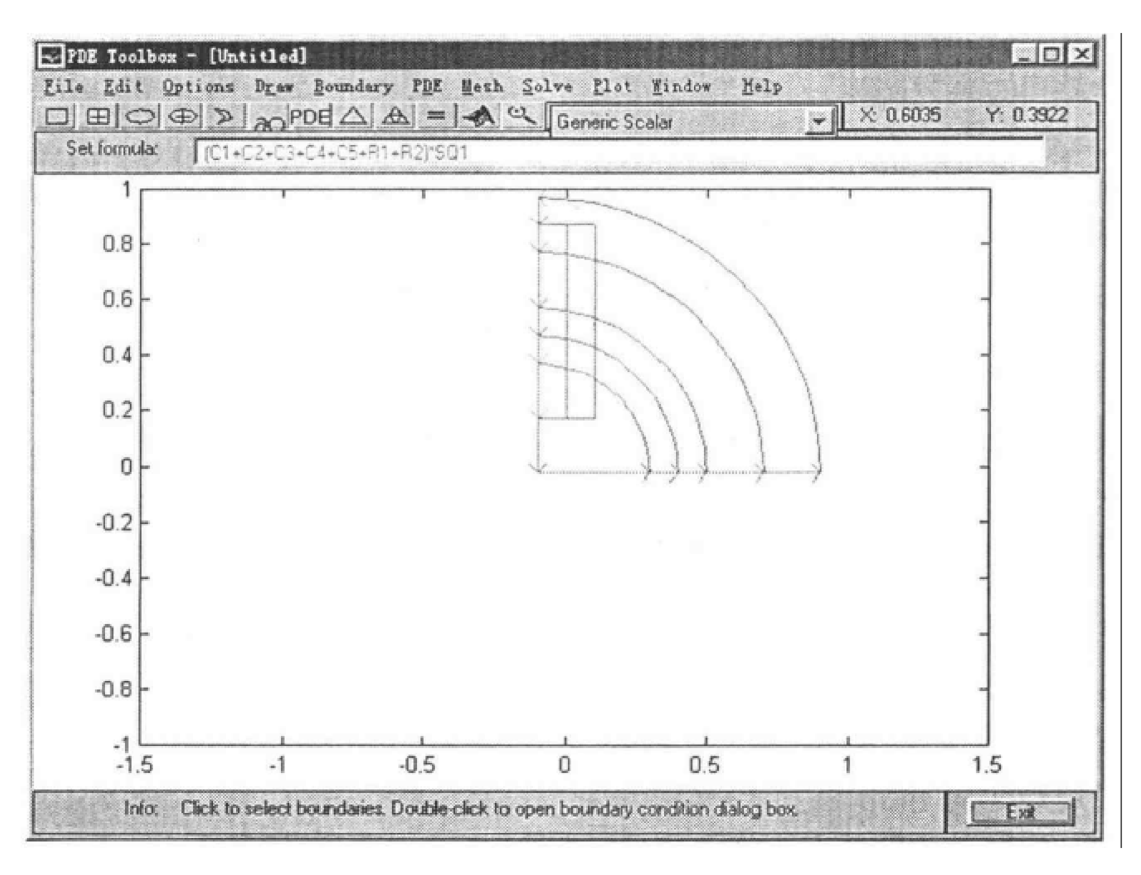


图7.3.5-4

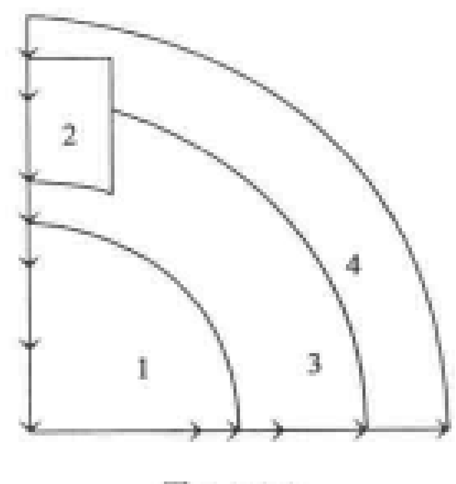


图 7.3.5-5

3. 边界条件设置：单击 ∂Ω ，使边界变红，然后分别双击边界，键入边界条件：x 轴边界选齐次 Neumann 条件，键入 g=0，q=0。其他边界选齐次 *Dirichlet* 条件，已默认这种边界条件。

4. 方程的设置：分别双击 4 个不同区域，设置方程的系数：线圈区域键入 μ=1，J=1。定子和转子区域键入 μ=5000/(1+0.05*(ux.^2+uy.^2))+200，J=0，这里系数含有非线性。空隙区域键入 μ=1，J=0。

5. 求解：在 Plot 中选定彩色、等值线、矢量线和磁流密度，对话框如图 7.3.5-6，立即可以求出这些量，如图 7.3.5-7 所示。

图 7.3.5-6



图 7.3.5-7

四、交流电磁场问题

在研究交流电的马达、变压器和导体问题时会遇到这一问题。考虑均匀电介质，介电系数为 ε，磁导率为 μ，不带任何电荷。这时电磁场服从一般的 Maxwell 方程组

$$\nabla\times\boldsymbol{E}=-\mu\frac{\partial\boldsymbol{H}}{\partial t}$$

$$\nabla\times\boldsymbol{H}=\dot{\varepsilon}\frac{\partial\boldsymbol{E}}{\partial t}+\boldsymbol{J}$$

在没有电流的情况下，可以从第一式消去 $\boldsymbol{H}$，第二式消去 $\boldsymbol{E}$，于是它们满足波速为$\sqrt{\varepsilon\mu}$的波动方程

$$\Delta\boldsymbol{E}-\varepsilon\mu\frac{\partial^2\boldsymbol{E}}{\partial t^2}=0$$

$$\Delta\boldsymbol{H}-\varepsilon\mu\frac{\partial^2\boldsymbol{H}}{\partial t^2}=0$$

进一步研究无电荷的均匀电介质情况，这时电介质系数为 ε，磁导率为 μ，电导率为 σ，于是电流密度为

$$\boldsymbol{J}=\sigma\boldsymbol{E}$$

这样就得到有阻尼的波动方程

$$\Delta\boldsymbol{E}-\mu\sigma\frac{\partial\boldsymbol{E}}{\partial t}-\varepsilon\mu\frac{\partial^2\boldsymbol{E}}{\partial t^2}=0$$

关于 $\boldsymbol{H}$ 也有类似的有阻尼的波动方程。

考虑频率一定的谐波情形，只需用复指数 $E_c e^{j\omega t}$ 代替上述式中的 $\boldsymbol{E}$($j=\sqrt{-1}$)，对于 PDE Toolbox 模型中的平面情形有 $\boldsymbol{E}_c=(0,0,E_c)$，$\boldsymbol{J}=(0,0,Je^{j\omega t})$，以及磁场

$$H=(H_x,H_y,0)=-\frac{1}{j\mu\omega}\nabla\times E_c$$

标量 E_c 的方程为

$$-\nabla\cdot\left(\frac{1}{\mu}E_c\right)+(j\omega\sigma-\omega^2\varepsilon)E_c=0$$

解这个方程需要用PDE Toolbox 中 AC Power Electromagnetics 模型。这是一个复的 Helmholty 方程，它描述在非理想介质和良导体($\sigma\gg\omega\varepsilon$)中平面电磁波的传播。复的介电系数 ε_c 定义为 $\varepsilon_c=\varepsilon-j\sigma/w$。在不同材料的交界面上 ε 和 μ 是不连续的，由于满足变分原理的自然边界条件，故不需要单独表示出来。

关于 PDE 参数需要在 PDE Specification 对话框中键入圆频率 ω，磁导率 μ，电导率 σ 以及电介质系数 ε。

这个模型的边界分 Dirichlet 条件和 Neumann 条件，后者给的是 E_c 的法向导数，也就是 H 的切向分量

$$H_t=\frac{j}{\omega}n\cdot\left(\frac{1}{\mu}\nabla E_c\right)$$

当解出电磁强度 E，可进一步计算电流密度 $J=\sigma E$ 以及磁感强度 $B=\frac{j}{\omega}\nabla\times E$，还可以作出场强 E，电流密度 J，磁场强度 H 和磁感强度 B 的图形。同时热阻率 $Q=E_c^2/\sigma$ 也可以作图，对于 H 和 B 也可以用矢量作出其向量场。

【例 7.3.5-4】 考虑圆截面线圈通过交流电流会产生趋肤效应。铜的电导率为 57×10^6，磁导率为 1，即 $\mu=4\pi\times10^{-17}$，在线频率 50HZ，$\omega^2\varepsilon$ 项可以忽略。

由于感应作用，导体内部的电流密度远远小于外表面的电流 $J_s=1$，这里电场的 Dirichlet 条件 $E_c=\frac{1}{\sigma}$。这种情况能够求出解析解

$$J=J_s\frac{J_0(kr)}{J_0(kR)}$$

其中 $k=\sqrt{j\omega\mu\sigma}$，$R$ 是线圈的半径，r 是离圆心的距离，$J_0(x)$是第一类零阶 Bessel 函数。使用 GUI 求解：

1. 在 PDE Toolbox GUI 中选应用模型 AC Power Electromagnetics。

2. 画图：双击圆中任一点，键入圆心(0,0)，半径 radius=0.1。

3. 边界条件设置：单击 Boundary，选 Boundary mode，再单击 Boundary，选 Specify Boundary conditions，在对话框中键入 Dirichlet 边界条件 $r=1/57E6$。

4. 方程设置：打开 PDE Specification 对话框，选 Elliptic，键入参数：Omega=2*pi*50，mu=4*pi*1E-7，sigma=57E6，epsilon=8.8E-12。

5. 网格生成：单击 △，加密网格单击[图标]。

6. 图形解设置：单击 plot，再单击 parameters，选 color，contour，show mesh，再单击 plot，由于解是复数，显示的只是其实部的彩色等值图(具有网格剖分)。选择 Height(3-Dplot)。

7. 求解：再单击 plot，作出解的彩色三维图形。

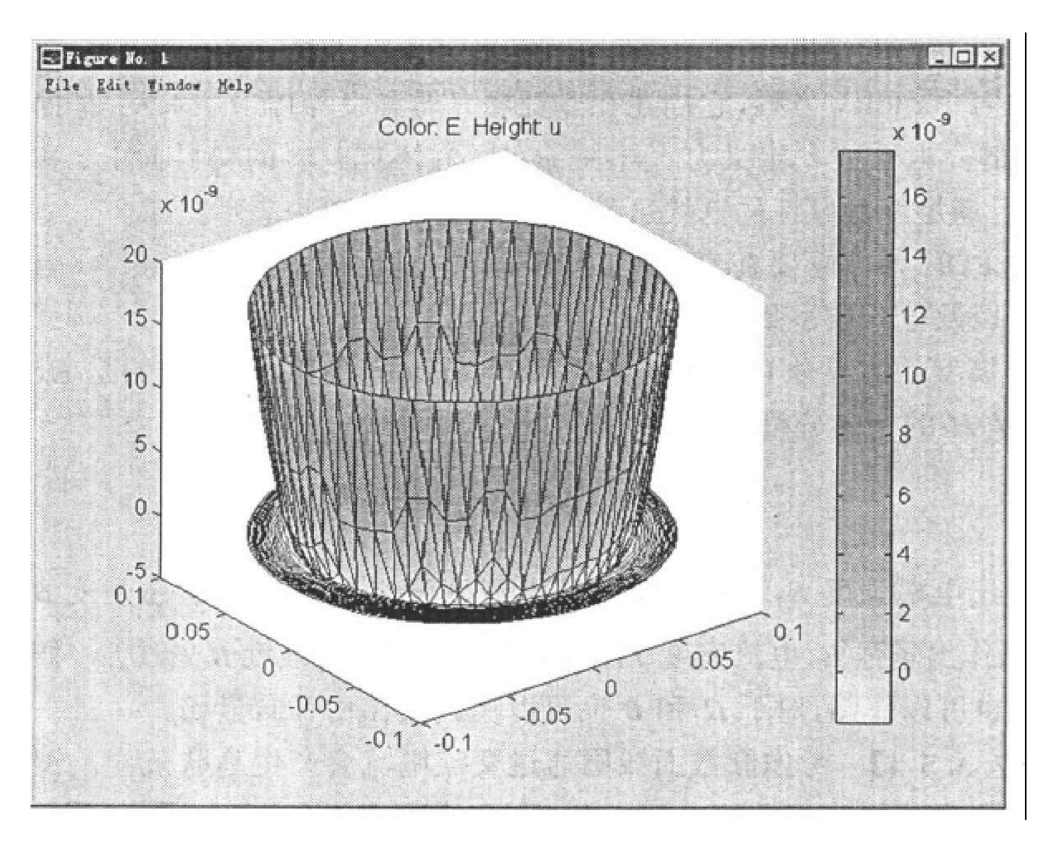


图 7.3.5-8

五、直流导电介质问题

考虑导电介质的电导率为 σ，在定常电流下，由电流密度 $\boldsymbol{J}$ 与电场强度 $\boldsymbol{E}$ 的关系 $\boldsymbol{J}=\sigma\boldsymbol{E}$，又由 $\nabla\cdot\boldsymbol{J}=Q$($Q$ 为电流源)，得到用电位 V 表示的 Poisson 方程

$$-\nabla\cdot(\sigma\nabla V)=Q$$

如果边界给 Dirichlet 条件，即在边界给出电位 V 的值；如果边界给 Neumann 条件，即在边界给出电流密度的法向分量($\boldsymbol{n}\cdot(\sigma\nabla(V))$)。也可以给出一般 Neumann 条件为

$$\boldsymbol{n}\cdot(\sigma\nabla(V))+qV=g$$

Toolbox 中可以作电位 V、电场强度 $\boldsymbol{E}$ 以及电流密度 $\boldsymbol{J}$ 的图形。如果 σ 是各向同性的，那么等电位线与电流方向处处垂直。

【例 7.3.5-5】 在一个矩形介质中有两个小矩形，这两个矩形均由金属导体

组成，其中一个矩形保持电位为1，另一个保持电位为−1，而大矩形外边界满足Neumann条件$\frac{\partial V}{\partial n}=0$，介质的电导率为1。

使用GUI求解：

1. 应用模式选择：在菜单或工具栏应用模型中选Conductive Media DC。

2. 画图：区域的设置作矩形R_1，其两个角点坐标为(−0.9,0.6)，(0.1，−0.4)。再作两个矩形R_2(−0.7,0.4)，(−0.5，−0.2)和R_3(−0.3,0.4)，(−0.1，−0.2)表示导体。在公式栏Set formula中输入$R_1-(R_2+R_3)$。

3. 边界条件：在Boundary Condition对话框中设置外边界为齐次Neumann条件，左右两矩形边界为Dirichlet条件，左边的为V=1，右边的为V=−1。

4. 方程参数：再打开PDE Specification对话框，设q=0，σ=1。

5. 网格剖分：初始化网格，并且加密两次。

6. 网格优化：还可以通过调整优化(jiggling)网格，使三角剖分更合理。

7. 求解：最后单击 = 按钮，得到电位的等值线。y轴上的电位为0，对于y轴点有反对称分布。可以作电流密度$\boldsymbol{J}$的等值线、矢量线。电流密度$\boldsymbol{J}$的等值线如图7.3.5-9所示。

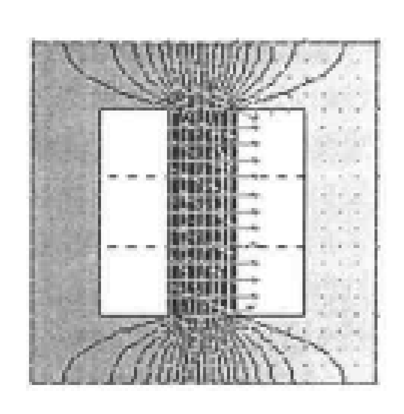


图 7.3.5-9

六、热的传输问题

这类问题属于抛物型PDE：

$$c\rho\frac{\partial T}{\partial t}-\nabla\cdot(k\nabla T)=Q+h\cdot(T_{ext}-T)$$

它描述了平面的热传输问题，或者是轴对称三维问题经过降维后的热传输问题，其中T为温度，其他参数为：

ρ—密度，c—比热，k—导热系数；

Q—热源，h—对流热的传输系数；

T_{ext}—环境温度，$h\cdot(T_{ext}-T)$表示从环境向区域内部的传输热量。

如果仅考虑定态情况，那么方程变为

$$-\nabla\cdot(k\nabla T)=Q+h\cdot(T_{ext}-T)$$

边界条件仍然可以有 Dirichlet 条件、Neumann 条件和一般 Neumann 条件。

在 Toolbox 中可以作温度、温度梯度、热流 $k\nabla T$ 的可视化图形。

下面是一个不同介质系数的热传输问题。

【例 7.3.5-6】 考虑一个方形区域，导热系数为 10，密度为 2；在方形区域内有一个圆形区域的热源为 4，导热系数为 2，密度为 1。两个区域的比热都为 0.1。

用 GUI 求解：

1. 应用方式设置：打开 PDE Toolbox GUI，在应用 mode 中选 Heat Transfer。

2. 绘图工具设置：选择 x 轴和 y 轴的范围为[−0.5,3.5]，然后从 Options 菜单中选 Axis Equal。

3. 绘图：方形区域的顶点为(0,0)，(4,4)。圆形区域的圆心为(2,2)，半径为 2。

4. 边界条件设置：所有外边界的温度保持为 0，所以不必修改缺省边界条件。

5. PDE 参数设置：进入 PDE mode，分别双击两个区域设置 PDE 参数。选择 Parabolic 方程后，对于矩形区域设定密度 rho 为 2，比热 C 为 0.1，导热系数 k 为 10，热源 Q 为 0。对于圆形区域设定密度 rho 为 1，比热 C 为 0.1，导热系数 k 为 2，热源 Q 为 4。因为导热项 $h\cdot(T_{ext}-T)$没有使用，故 h=0。

6. 解的参数设置：因为要求的解是随时间变化的，故需要定义初始值和时间间隔。打开 Solve Paramentes 对话框。这一问题的动态变化非常快，温度在大约 0.1 单位时间即趋于定常。为了捕捉动态的有兴趣部分，只需输入时间向量 logspace(−2,−1,10)，它可在 0.01 和 0.1 之间等分 10 个对数单位。

7. 温度的动态性质可视化的最好办法就是解的动画。转入 Height(3-Dplot)选 3-D 动画作图。图 7.3.5-10 给出 t=1 时温度 T 和−grad(T)的图形。

七、扩 散 问 题

因为热的传输是一个扩散过程，因此一般的扩散方程也与热传导方程相同：

$\frac{\partial c}{\partial t}-\nabla(D\nabla c)=Q$，其中 c 是浓度，D 是扩散系数，Q 是源项。

扩散过程可以是各向异性的，只要取 D 为 2×2 的矩阵。边界条件可以是 Dirichlet 条件，或者是 Neumann 条件。浓度、浓度梯度、热流的可视化可以从 Plot Selection 对话框中实现。

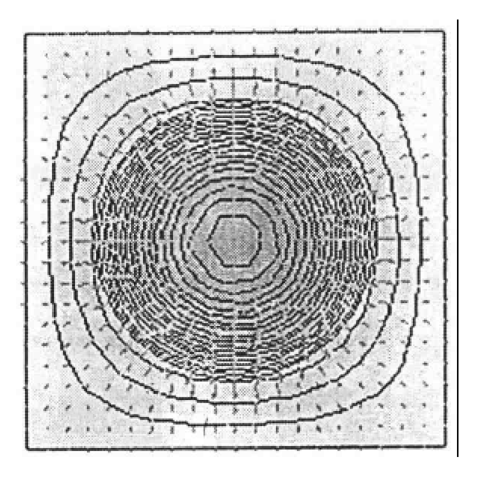


图 7.3.5-10

习　题　七

1. 按下列步骤计算描述一个区域内的点力，并检验是否与结果相符。

(1)画一块区域、一个圆点(不需设边界条件)(如图 1)。

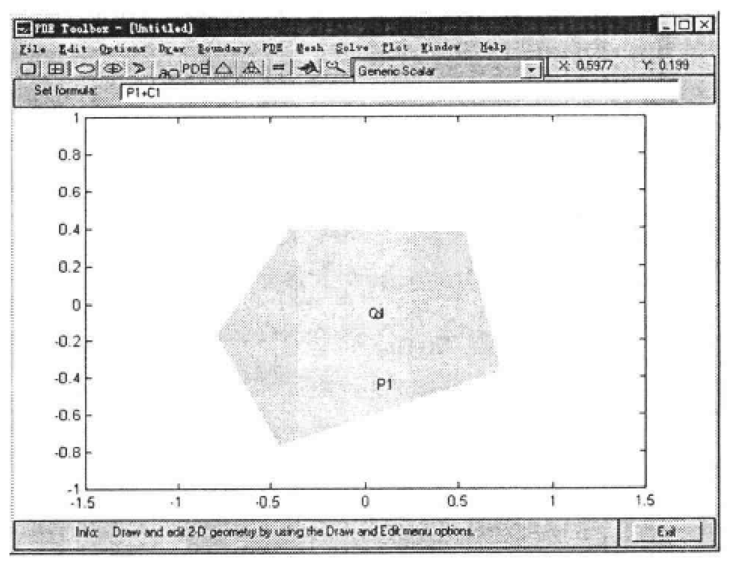


图 1

(2)设方程条件：

先进入 PDE 模式，双击区域：输入 c＝1，a＝0，f＝0；双击圆点：输入 c＝1，a＝0，f＝20(力的大小)。

(3)进行网格剖分(如图2)。

(4)解方程(如图3)。

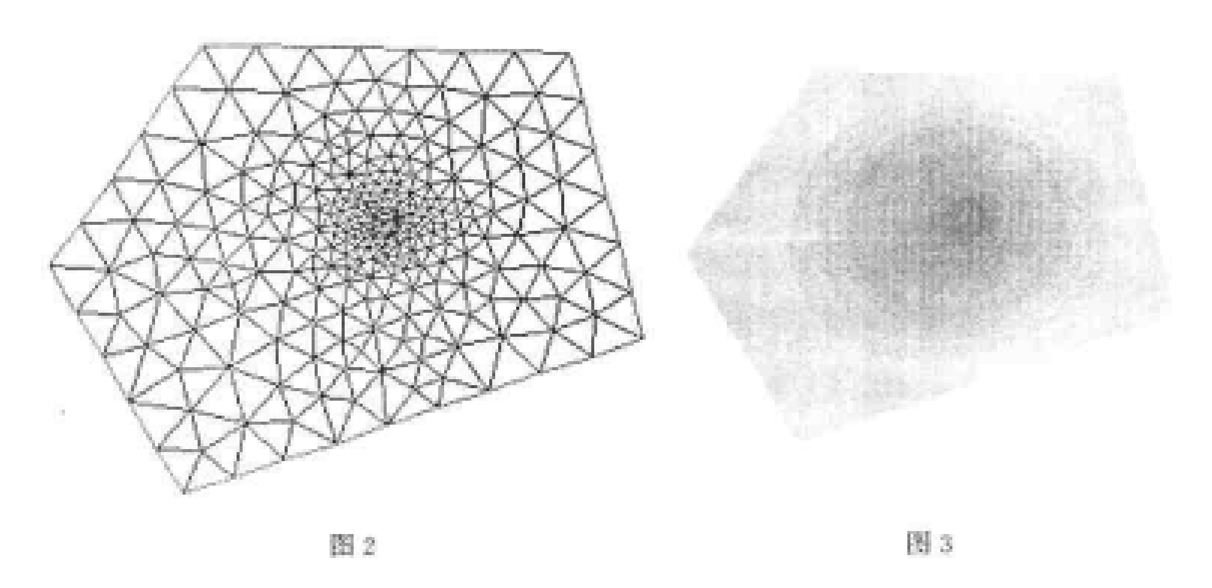


图2　　图3

(5)画三维图(结果如图4)。

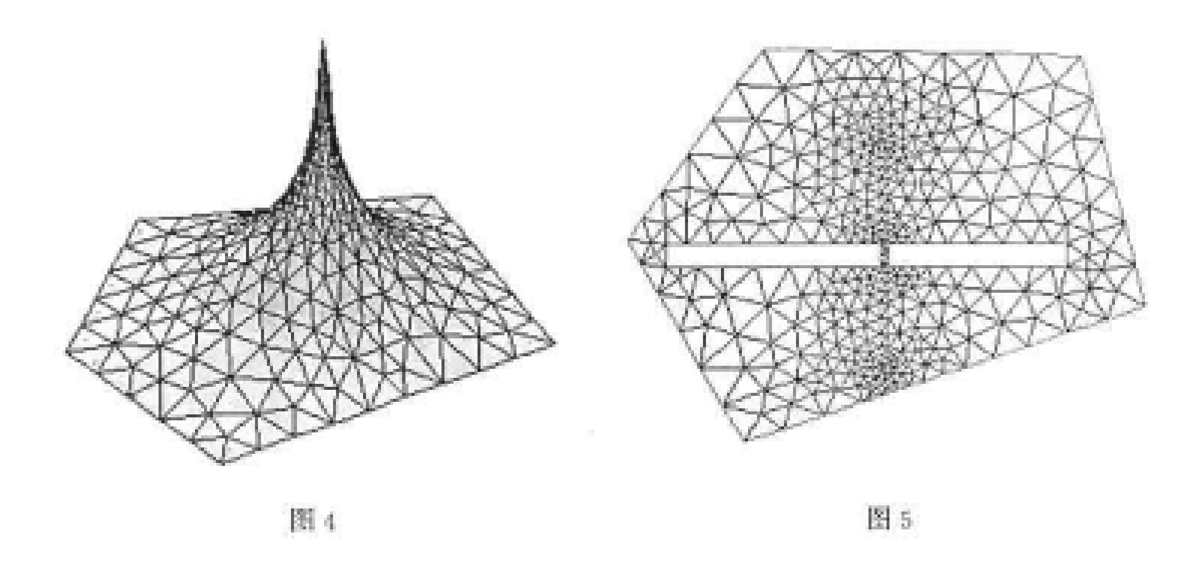


图4　　图5

2. 按下列步骤计算，描述一个区域内的裂缝和条状分布力，并检验是否与结果相符。

(1)画一块区域和两个矩形垂直相交，一个是裂缝，另一个是条状分布力。

(2)裂缝不需设方程条件，但设边界条件(如neumann条件为0)。

(3)分布力：设方程条件(不设边界条件)：c=1；a=0；f=20。

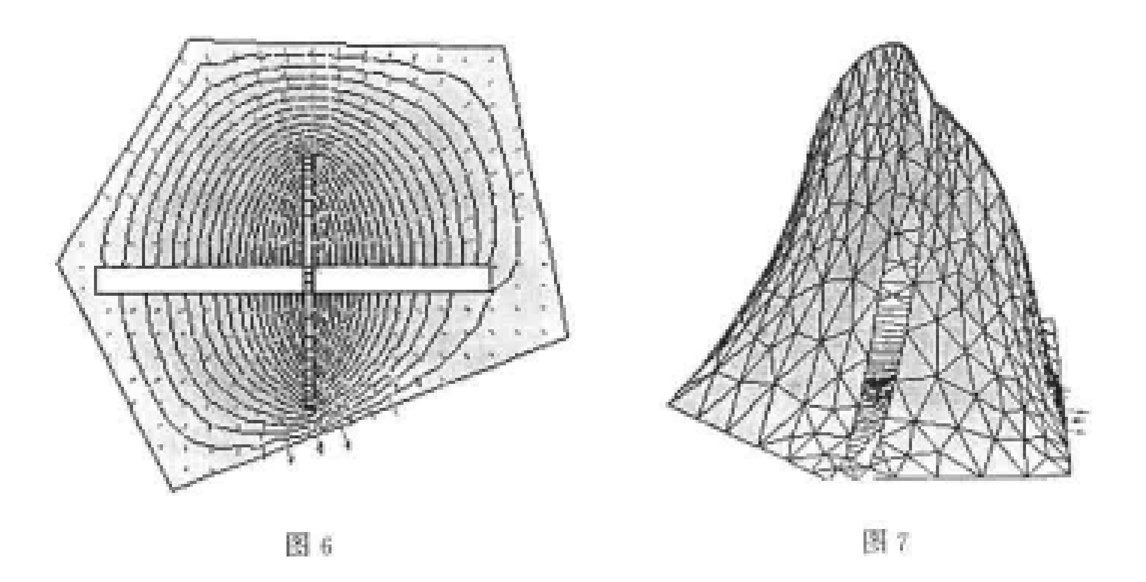


图 6　　　　图 7

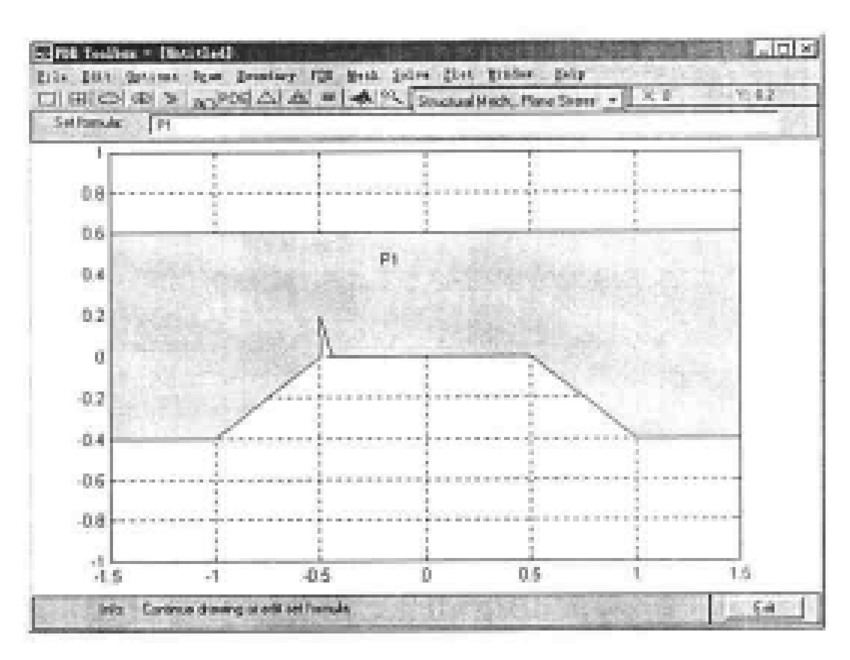


图 8

区域；设方程条件(不设边界条件)：c＝1；a＝0；f＝0。

(4)进行网格剖分(如图 5)。

(5)解方程(如图 6)。

(6)画三维图(结果如图 7)。

3. 一块带缺口钢板，下方两边被钳住，其上方的一边受拉力，其他边均自由。

假设钢板厚为 1mm，钳住的边皆为 0.12m，钢板尺寸如图 8 所示。杨氏模量为 196 $\times 10^3$(MN/m^2)，Poisson 比为 0.31。圆弧边界受外法向荷载为 500N/m，由于钢板厚为 1mm，所以钢板表面牵引力为 0.5MN/m^2。

步骤如下：

(一)弹出 Application Mode，选择 Structural Mechanics，Plane Stress。

(二)区域设置：如图 8 所示。

(三)边界条件设置：单击 Boundary，下面的两边上是固定的，即满足齐次 Dirichlet 条件，上方边界上满足 Neumann 条件：q=0，g1=0.5*nx，g2=0.5*ny，其余边界是自由的，即满足齐次 Neumann 条件：q=0，g=0。

(四)方程设置：打开 PDE Specification 对话框，键入参数：杨氏模量 E=196E3，Poisson 比 v=0.31，因为没有受力，故 K_x 和 K_y 为 0，而 ρ(rh0)在本例中不使用。

(五)网格剖分：单击 △ ，再单击 △ 加密。

(六)求解：单击 Plot，再单击 Plot Selection，在菜单 Plot Parameter 下生成的对话框 Plot Selection 中设置输出各种变量的图形，如 u 和 v 方向位移，x 和 y 方向应变、应力、剪切应力、剪切应变、von Mises 等效应力以及主应力、应变等等。利用变形网格可以作出 von Mises 有效应力和位移图形，如图 9 所示，如果上边界受压力，再试求一下。

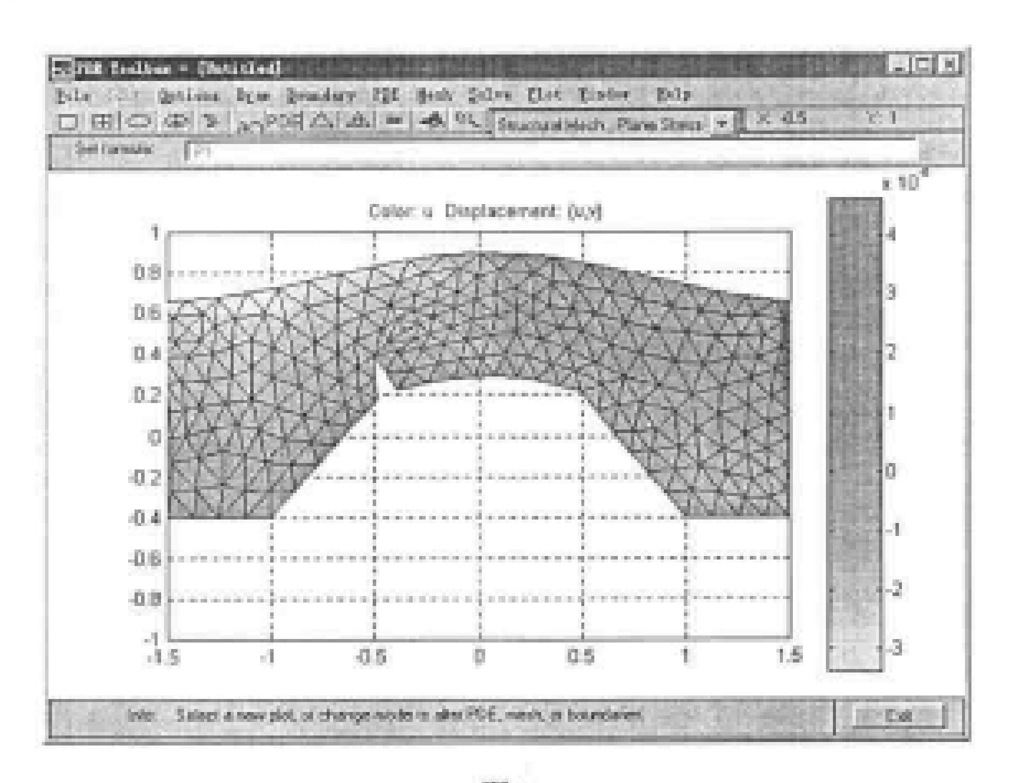


图 9

第八章　优化设计

8.1　引例与数学模型

在工程设计中，常常遇到多约束求极值或对局部及整体进行优化等问题。

现在举一个线性归化问题：

某工厂生产A，B两种产品，所有原料均为甲、乙、丙三种，每生产一个产品具体用料如下：

	甲	乙	丙	利润
A	9	3	4	7千元
B	4	10	5	12千元
用料	360	300	200	

已知产品A每件可得利润7千元，产品B可得利润12千元，如果这个工厂只有原料甲360单位，原料乙300单位，原料丙200单位，问在这种条件下，应该生产A，B产品各多少才能使获得的利润最多？

设生产A，B产品为x_1，x_2个，则所获利润为

目标函数：　$z=7x_1+12x_2$

约束条件：　$9x_1+4x_2\leqslant 360$

$3x_1+10x_2\leqslant 300$

$4x_1+5x_2\leqslant 200$

算法介绍

1. 一维搜索方法

基本思想是利用目标函数$f(x)$在某些点的信息，构造一个近似函数$p(x)$，用$p(x)$的极小点作为函数$f(x)$极小点的极小值。显然$p(x)$应尽量简单。一般可选为二次函数、三次函数或有理函数，分别称为二次插值、三次插值、有理插值。

(一)抛物线插值法(三点二次插值)

取$p(x)$为二次函数，需用三个条件来确定$p(x)$的待定系数。假定目标函数$f(x)$在区间$[a,b]$上有惟一极小点x^*，并有三个点x_1，x_2，x_3及其函数值$f(x_1)$，$f(x_2)$，$f(x_3)$，满足

$$x_1<x_2<x_3$$

$$f(x_1) > f(x_2) < f(x_3)$$

作二次函数 $p(x)$:

$$p(x)=a_0+a_1x+a_2x^2$$

使 $p(x)$ 在点 x_1,x_2,x_3 处的函数值分别等于 $f(x_1),f(x_2),f(x_3)$，然后求 $p(x)$ 的极小点 x_4。以 x_4 作为 $f(x)$ 的极小点 x^* 的近似值，误差可能太大，故以 x_4 作为插值节点，根据不同情形，去掉点 x_1 或 x_3，得三个新的节点，并缩短了搜索区间，重新插值，又得到新的极小点以逼近 x^*，如此反复进行，直到求得的极小值满足给定的精度要求为止。

(二)两点三次插值法

两点三次插值法与抛物线插值法相似，只是三次插值法是用三次函数 $p(x)$ 作为 $f(x)$ 的四个待定系数。为此假定 $f(x)$ 在搜索区间 $[a,b]$ 上连续可导，x^* 是 $f(x)$ 在 $[a,b]$ 上的惟一极小点，且 $f'(a)<0,f'(b)>0$，以 a,b 为插值节点作三次函数 $p(x)$ 使

$$p(a)=f(a) \qquad p(b)=f(b)$$

$$p'(a)=f'(a) \qquad p'(b)=f'(b)$$

求 $p(x)$ 的极值点 $\tilde{x}$，以 $\tilde{x}$ 作为 x^* 的近似值。如果误差较大，则检验 $f'(x)$ 的符号；如果 $f'(\tilde{x})<0$，则取 $\tilde{x},b$ 为新的插值节点，令 $a=\tilde{x}$；如果 $f'(\tilde{x})>0$，则取 $a,\tilde{x}$ 为新的插值节点，令 $b=\tilde{x}$，再进行三次插值，如此反复迭代，直至求得的 $p(x)$ 极小点 $\tilde{x}$ 满足给定的精度要求为止。

2. 无约束最优化的梯度方法

该方法可分两类：

一类是要利用目标函数的梯度(一阶导数或 Hessain 阵(二阶导数))所提供的信息而构造出来的迭代方法，统称梯度方法，它分为：最速下降法；Newton 法；共轭梯度法；变尺度法。

另一类是不利用导数的方法，称为直接方法，它分为：模式搜索法；单纯形法；powell 方法。

前者收敛速度较快，但要梯度与 Hessain 阵，计算量较大；后者不必计算导数，适应性较强，计算量较小，收敛速度较慢。

最速下降法也叫梯度法。它是根据目标函数 $f(x)$ 在迭代点 $x^{(k)}$ 处的最速下降方向是负梯度方向这一特性，从而选择

$$p(k)=-\nabla f(x^{(k)})=-g^{(k)}$$

作为搜索方向，来确定逐次搜索最优解 x^* 的方向。

8.2　优化工具箱

优化工具箱(Optimization Toolbox)的所有函数被放在 toolbox 目录下的 Optim 子目录中，涉及函数的最小化或最大化问题，也就是函数的极值问题。MATLAB 的优化工具箱包含：普通非线性函数求解最小化或最大化极值的函数、求解诸如线性规划等标准矩阵问题的函数等。

优化工具箱中求非线性函数极小值的函数如表 8-1 所示。

表 8-1

类　型	含　义	函数用法
无限定条件标量问题	$\min_x f(x)$，其中 x 为标量	x=fmin('f',x)
无限定条件矩阵问题	$\min_x f(X)$，其中 X 为矩阵	x=fminu('f',x)
有限定条件	$\min_x \min f(x)$，条件为 $G(x)\leqslant 0$	x=constr('fg',x)
多目标优化	$\min_x \gamma$，条件为 $F(X)-W^{T}\leqslant goal$	x=attgoal('f',x,goal,w)
最小最大极值	$\min_x \min\{\max F(X)\}$，条件为 $G(X)\leqslant 0$	x=minimax('fg',x)
非线性二次平方极值	$\min \sum (F(X)\cdot F(X))$	x=leastsq('f',x)
非线性方程	$F(X)=0$	x=fsolvex('f',x)
半无穷条件	$\min_x f(x)$，条件为 $\phi(X,\omega)\leqslant 0,\forall\ \omega$	x=seminf('ft',n,x)

上述运算对标量、向量、矩阵均可进行，矩阵用大写字母表示，向量、标量用小写字母表示。在 MATLAB 中，符号“ * ”表示矩阵的元素乘。上述运算要求事先定义一个要进行最小化的函数。

利用优化工具箱进行极值运算时，可以自由选择算法和线性搜索策略。无约束最小问题的原理是 Nelder-Mead 的单纯形方法和 BFGS 拟牛顿(Quasi-Newton)方法；约束条件下的最小、最小化最大、目标法和半无穷优化等问题，所用的原理算法是二次规划法；非线性二次平方问题的原理算法是 Gauss-Newton 法和 Levenberg-Marquardt 法；非线性最小和非线性二次平方问题，可选择线性搜索，线性搜索使用的是三次或四次内插和外插方法。

优化工具箱还能解决几类求矩阵的极小值问题，此时仅需要将相应的系数矩

阵和向量传递到函数中。MATLAB 能解决的矩阵问题如表 8-2 所示。

表 8-2

类　型	含　义	函数用法
非负二次平方问题	$\min\limits_x \| Ax-b \|^2$,条件为 $x\geqslant 0$	x=nnls(A,b)
二次问题	$\min\limits_x(1/2x^THx-c^Tx)$,条件为 $Ax\leqslant b$	x=qp(H,c,A,b)
线性规划问题	$\min\limits_x(f^Tx)$,条件为 $Ax\leqslant b$	X=lp(f,A,b)

读者可以运行 optdemo 的一个 m 文件进行优化工具箱的演示。tutdemo.m 提供了一个使用优化工具箱的指导示例,它包括了上述所有优化问题。下面通过例子详细说明。

8.3 优化设计实例分析

8.3.1 无约束极值问题

求合适的集合$[x_1,x_2]$,使

$$\underset{x}{\text{minimize}}\ f(x)e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$$

成立。

编程时,先编写一个能返回函数值的 m 文件,将函数表达式写入 MATLAB 环境中,然后调用无约束最小程序 fminu。具体过程如下:

第一步:利用文件编辑器编写 m 文件,即名为 fun 的函数。

```
function    f=fun(x)
f=exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

第二步:在命令窗口中调用优化程序。

```
x0=[-1,1];     %估计初值
x=fminu('fun',x0)
```

运行结果,函数迭代计算 36 次后,输出极值点为 x=[0.5000 -1.000]。

在命令窗口中接着键入 fun(x),则得到极值点处的函数值为 ans=1.3030e-010。

当函数存在一个以上的局部最小点时,$[x_1,x_2]$的初值会影响到迭代次数的解的数值。在上述例子中,x 的初值为[-1,1]。

为了改变优化的特征,可在函数 fminu 的说明参数中加入可选的参数说明,为

```
x=fminu('fun', x,options);
```

options 是一个包含允许误差和算法选择项的向量,其功能参见 8.4 节表 8-3。

在上述无约束极值问题中,改变选项 options(2)和 options(3)来提高求解的精度。程序如下:

```
x0=[-1,1];          %估计初值
options(2)=1e-8; %x 的中断允许误差
options(3)=1e-8; %f 的中断允许误差
[x,options]= fminu('fun', x0,options);%对函数进行极小化
x                   %输入极值点
options(8)          %输出极值点的函数值
options(10)         %输出函数计算的总次数
```

运行程序,输出结果为:

x=[0.50000,-1.0000];

极值点处的函数值:3.4412e-014;

函数计算的总次数:61 次。

若想以列表形式显示每一次迭代计算的结果,可以利用 options(1)=1。程序如下:

```
options(1)=1;        %显示每一次迭代计算结果
x0=[-1,1];           %初始估计值
[xx,options]=fminu('fun',x0,options);     % 对函数进行极小化
x                    %输出极值点
options(8);          %输出极值点的函数值
options(10);         %输入函数迭代次数
```

运行结果为:

f-COUNT	FUNCTION	STEP-SIZE	GRAD/SD LINE-SEARCH
4	1.8394	1	-0.677
9	1.72427	0.361834	-0.00354 incstep
17	0.362233	5.90885	-0.295 incst2
20	0.221775	3	0.436 int _ st
25	0.0171251	0.529597	0.00204
30	0.0000631009	0.25069	-0.0000292 incstep
36	6.2144e-008	1.37782	1.27e-009 int _ st

```
Optimization Terminated Successfully
Gradient less than options(2)
NO OF ITERATIONS=36
X=[0.50000 -1.0000]
```

极值点函数值为1.3030e-010；

函数迭代次数为36。

8.3.2 约束极值问题

对于约束极值问题，使用constr函数，在上述8.3.1无约束极值问题中加入不等式约束条件，例如：

$$\text{minimize} f(x)=e^{x_1}(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)$$

约束条件：$1.5+x_1x_2-x_1-x_2\leqslant 0$

$-x_1x_2-10\leqslant 0$

将上面编写的m文件进行修改，使其返回目标函数和约束函数值，然后应用约束条件下的极值函数constr求出极值，具体过程如下：

第一步：利用文件编辑器编写m文件。

```
function [f,g]=fun(x)
f=exp(x(1))*(4*x(1)^ 2+2*x(2)^ 2+4*x(1)*x(2)+2*x(2)+1);
g(1)=1.5+x(2)-x(1)-x(2);
g(2)=-x(1)*x(2)-10;
```

第二步：在命令窗口中调用优化程序。

```
x0=[-1,1];            %估计初值
options=[];           %使用缺省参数
[x0,options]=constr('fun',x0,options)
[f,g] =fun(x)         %输出极值点的函数值及约束条件值
options(10)           %输出函数计算次数
```

运行后得到极值点x=[-9.5471　1.0474]；极值点处的函数值f=0.0236；约束条件值g=1.0e-1014*[0.1110;-0.1776]；函数计算次数为29。

此问题比较简单，可直接编写一个m文件运行，如

```
x0=[-1,1];            %首先给出极值点的估计值
options=[];           %给出参数项，使用缺省参数
funf='f=exp(x(1))*(4*x(1)^ 2+2*x(2)^ 2+4*x(1)*x(2)+2*x(2)+1)';
fung='[1.5+x(1)*x(2)-x(1)-x(2);-x(1)*x(2)-10]';
fun=[funf,fung];
[x,options]=constr('fun',x0,options);        %对函数进行极小化
```

```
x                   %输出极值点
options(8)          %输出在极值点的函数值
options(10)         %输出函数计算次数
```

8.3.3　上界条件和下界条件

使用函数的有界语法可将变量 x 限定在某一区域之内，如对于函数 constr 语句：

x=constr('fun',x,options,vlb,vub);

将 x 限定在 vlb<x<vub 范围内。

在上述 8.3.2 约束极值问题中，将 x 限定为大于 0，即第二步在命令窗口中输入：

```
x0=[-1,1];          %估计初值
vlb=[0.0];          %下界
vub=[];             %无上界
[x,options]=constr('fun',x0,options,vlb,vub);
[f,g]=fun(x)        %输出极值点的函数值及约束条件值
options(10)         %输出函数计算次数
```

运行后，输出 x=0　1.5000；极值点的函数值 f=8.5；约束条件值 g=0；函数计算次数为 7 次。

上例中，x 无上界，因此 vub 量为空矩阵，也可以使用如下形式的命令省掉上界：

[x,options]=constr('fun',x0,options,vlb)

当 vlb 或 vub 的元素数目比向量 x 的元素数目少时，则 x 中只有前几个元素被限定有界。另外，界也可以用线性不等式表示。当仅有几个元素有界时：

上界：$x_i \leqslant u_B$ 表示为 $x_i - u_B \leqslant 0$；

下界：$x_i \geqslant u_B$ 表示为 $-x_i + u_B \leqslant 0$。

作为约束条件处理，相应对 fun 函数加以修正。

8.3.4　梯 度 计 算

一般来说，最小化程序都要运用有限微分近似法计算梯度，这一过程会自动对每个变量计算函数和约束条件的偏导数。若所编写的程序能返回函数和约束条件的偏导数，则问题的求解将会更精确，更有效。

下面的例子是用梯度来解决前述约束极值问题，具体过程如下：

第一步：利用文件编辑器为目标函数编写 m 文件。

```
function  [f,g]=fun(x)
```

```
f=exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
g(1)=1.5+x(1)*x(2)-x(1)-x(2);
g(2)=-x(1)*x(2)-10;
```

第二步：编写求梯度的 m 文件。

```
function    [df,dg]=grad(x)
t=exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
df=[t+4*exp(x(1))*(2*x(1)+x(2)),
4*exp(x(1))*9x(1)+x(2)+0.5)];  %定义目标函数的梯度函数
dg=[x(2)-1,-x(2);x(1)-1,-x(1)];  %定义约束函数的梯度
```

第三步：在命令窗口中调用求约束条件下的极值问题的程序。

```
x0=[-1,1];          %初始估计值
options=[];         %用缺省选项
vlb=[];             %无下界
vub=[];             %无上界
options(9)=1;       %检查梯度错误
[x,options]=constr('fun',x0,option,vlb,vub,'grad')
```

其中变量 df 包含目标函数 fun(x)对于 x 中每一个元素的偏导数；变量 dg 的列向量包含对每一个约束条件的偏导数。

运行程序，经过 11 次函数计算和梯度计算后，得到的极值为 x＝－9.5474，1.0474。

程序中采用 options(9)＝1，目的是检查梯度计算是否出错。在优化检测的第一个周期中会自动检查梯度，若它们与给定的允许误差不匹配，将会出现警告信息，指出偏差大小，并给出是放弃优化还是继续优化的选择项。

8.3.5 最大值问题

优化函数 fminu，constr，attgoal，minimax 和 leastsq 都是执行目标函数 $f(x)$的极小值的函数，若要求极大值，可对$-f(x)$进行求极小值的运算，所得的结果即为极大值。

8.3.6 大于零的约束条件

优化工具箱只使用形如 $g_i(x)\leqslant 0$ 的约束条件，对于大于零的约束条件用乘以－1 的处理。例如形式 $g_i(x)\geqslant 0$ 等价于$-g_i(x)\leqslant 0$。

8.3.7 等式约束条件

等式约束条件式要放在矩阵 g 的前面位置，此时 options(13)所赋初值为条件

等式的个数。例如对 8.3.2 的约束极值问题增加 $x_1+x_2=0$ 的约束条件，可进行如下操作：

第一步：利用文件编辑器为目标函数编写 m 文件。

```
function[f,g]=fun(x)
f=exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
g(1)=x(1)+x(2);                        %等式约束条件
g(2)=1.5+x(1)*x(2)-x(1)-x(2);          %不等式约束条件
g(3)=-x(1)*x(2)-10;                    %不等式约束条件
```

第二步：在命令窗口中调用限定条件下的最优程序。

```
x0=[-1 1];               %初始估计值
options(13)=1;           %指出在限定条件中有一个等式约束条件
[x,options]=constr('fun',x0,options);
[f,g]=fu(x);
options(10);
```

运行程序，经过 11 次迭代后，得极值 x=-1.2247,1.2247；

极值点处的函数值为 f=1.8951；约束条件值为 g=[0.0000；-0.0000；-8.5000]。

8.3.8 参数传递

若需将参数定义为全局变量而直接传递到 m 文件中，可在调用语句的最后使用相应的附加说明，要注意的是被传递到文件中的最大参数数目为 10。例如在调用函数 fsolve 的结尾加入一些变量：

```
fsolve('fun',x0,options,gradfun,p1,p2)
```

在每一次迭代中，可直接应用上面的参数，例如可在函数 fun 和 grad 中应用：

```
f=fun(x,p1,p2,…)
dg=grad(x,p1,p2,…)
```

下面是一个具体的例子。要找出函数 ellipj(u,m)的零值，函数需要的参数为 m 和 u。对 m=0.5 寻找在 u=3 附近的零值，可用如下的语句实现：

```
x=fsolve('ellipj',3,[],[],0.5)
x                    %输入 x 值
f=ellipj(x,0.5)      %输出对函数 ellipj 的解
```

运行结果为：x=3.7081，f=6.2698e-12。

在 fsolve 语句中，方括号表示使用缺省选项，且不用梯度计算。

8.3.9 Banana 函数最小化演示程序

程序 bandemo.m 演示 Banana 函数（也称为 Rosenbrock 函数）的最小优化过

程，即求

$$f(x)=100(x_2-x_1^2)^2+(1-x_1)^2$$

的极值问题。

因为其收敛很慢，所以对大部分方法来说，是一个极难优化的例子。此函数在点 $x=[1,1]$ 处有惟一的最小值 $f(x)=0$。本示例程序在初始值为 $x=[-1.9,2]$ 的条件下演示用不同的方法进行最小化的过程。

首先画出函数的三维图形：

```
disp('trike any key for a mesh plot of the banana function')
xx=[-2:0.125:2]';
yy=[-1:0.125:3]';
[x,y]=meshgrid(xx',yy');
meshd=100.*(y-x.*x).^2+(1-x).^2;
pause
mesh(meshed)
```

所得的图形如图 8.3.9-1 所示。由图 8.3.9-1 可以看出，函数具有惟一的极值点。

下面的语句画出函数的等值线图形：

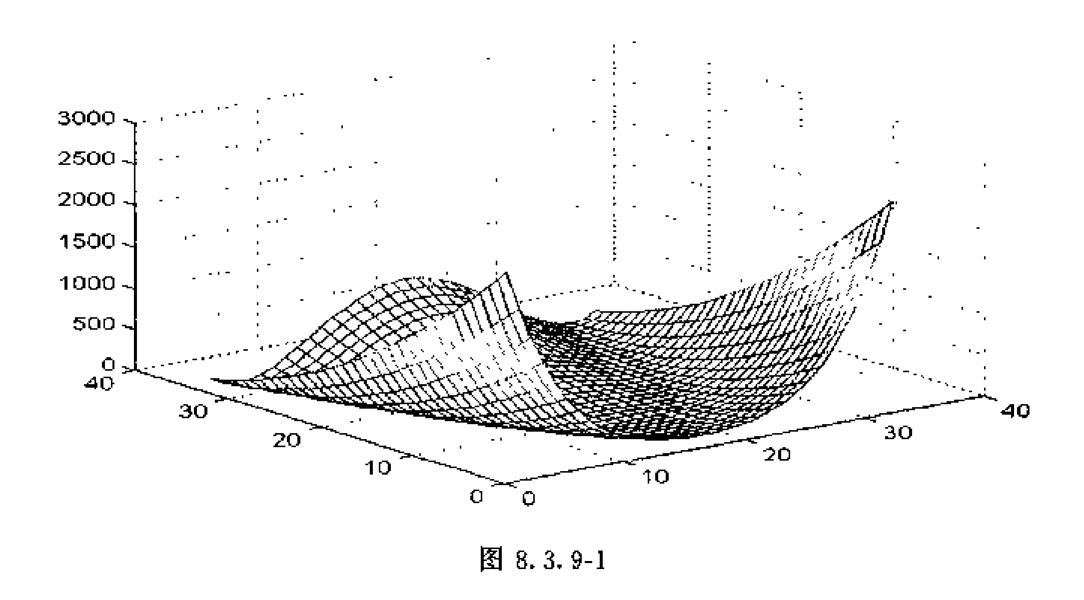


图 8.3.9-1

```
disp('')
disp('Strike any key for a contour plot of the banana function')
drawnow;  %画出当前图形
hold on
plot(-1.9,2,'o')
```

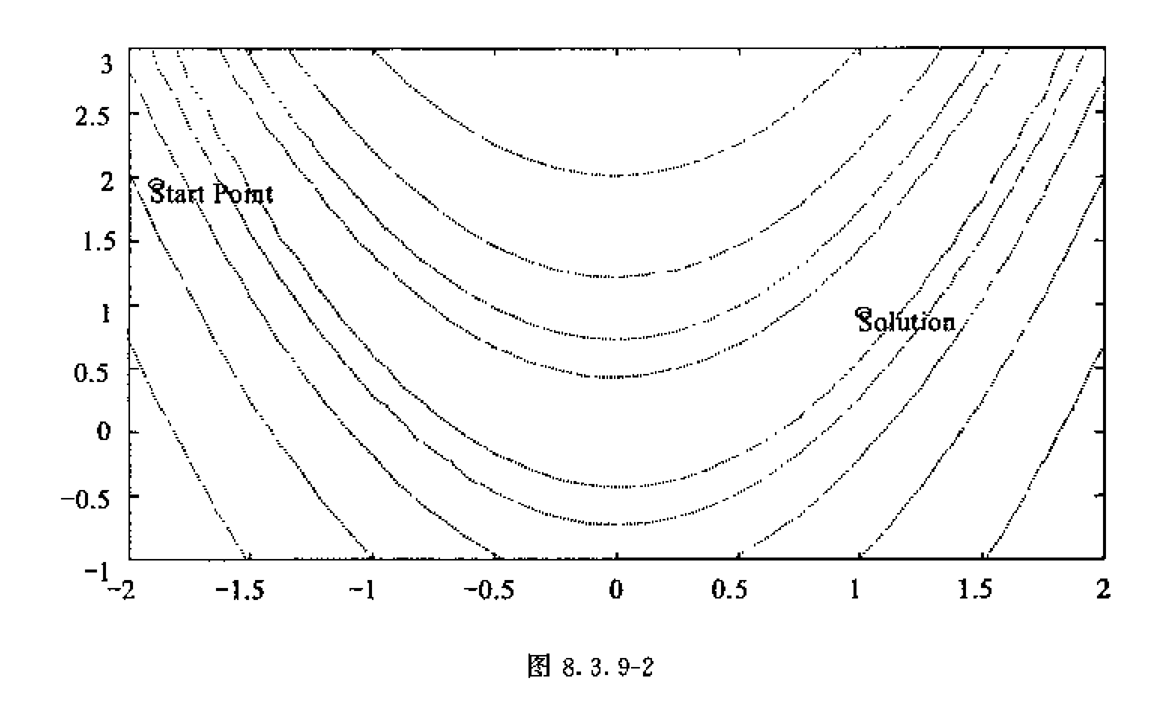


图 8.3.9-2

```
text(-1.9,2,'Start Point')
plot(1,1,'o')
text(1,1,'Solution')
```

其中函数 contour 画出函数的等值线图，其具体的使用方法可参阅 MATLAB 提供的帮助。在图形中用符号“o”分别标出了初始点 $x=[-1.9,2]$和极值点 $x=[1,1]$。接上述语句得到的图形如图 8.3.9-2 所示。

本程序用不同的方法对 banana 函数进行了极值运算，在命令窗口选择不同的方法可得到最小化结果。

例如选择 Broyden-Fletcher-Golfarb-Shanno 方法，并采用混合多项式内插线性搜索法，在出现菜单后，可进行如下的操作：

Select a method number:1

Choose any of the following line search methods

1) Mixed Polynomial Interpolation

2) Cubic Interpolation

Select a line search number:1

得到的结果为：

[x,options]=fminu(f,x,OPTIONS,GRAD)

在极值点处的函数值(Value of the function at the solution):8.03777e-015;

函数计算的次数(Number of function evaluations):85;

梯度计算的次数(Number of gradient evaluations):26。

8.3.10 表达式优化

优化工具箱中的函数还可直接对表达式优化,这样可避免编写定义函数的m文件,将表达式直接放在字符串中。若被计算的函数变量(例如fun)是包含非希腊字母及非数字的字符(例如*,-,+),它将以一个表达式而不是以函数进行计算。

书写这样的表达式时,自变量必须以小写字母x表示。下面是几个例子。

1. 求函数的极小值

```
x=fminu('sin(x)',1)   %求sin(x)的最小值,初值为1
```

该语句等价于fminu('sin',1)。

2. 平方问题

```
x=fminu('x(1)^2+x(2)^2',[1;1])
```

或用偏导数求解:

```
x=fminu('x(1)^2+x(2)^2',[1;1],[],'[2*x(1);2*x(2)]')
```

3. 矩阵等式

```
x=fsolve('x*x*x-[1,2;3,4]',zeros(2,2))
```

4. 最小平方问题

```
x=leastsq('x*x-magic(2)',eye(2,2))
```

5. 具有变量名的函数参数

```
x=attgoal('sort(eig(p1+p2*x*p3))',zeros(1;1),…
[-5,-3,-1],[],[],-4*ones(2),[],A,B,C);
```

此处的函数参数p1,p2和p3分别被置为等于变量A,B和C。

6. 具有约束条件的优化问题

```
x=minimax('f=x*x*x-[1,2;3,4];g=[];',ones(2));
x=constr('f=x(1)^2+x(2)^2;g=[];',[1;1],[1;1]);
```

上面两例中没有约束条件,因此矩阵g为空矩阵。对于8.3.2的约束极值问题,可写成

```
x=constr('f=exp(x(x))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+
      2*x(2)+1);g=[1.5+x(1)*x(2)-x(1) -x(2);-x(1)*x(2)-
10];',x0,options,vlb,vub);
```

8.3.11 常见问题及推荐的解决办法

优化问题可能要经过很多次迭代才能收敛,也可能对诸如有微分梯度计算中的截短或截断误差很敏感。大部分优化问题都需要有一个好的初始估计值,这能提

高执行的效率，并有利于发现全局最小而非局部最小。

运用优化工具箱可解决大量的问题，并能克服大部分与优化技术有关的局限性。另外，对非标准形问题，经过合适的变换后也能用本工具箱求解。下面是一些常见的问题和推荐的解决办法。

问题1：解不是全局最小。

推荐：除非问题是连续的且只有一个最小值，否则存在全局最小，这一点并不能得到保证。从不同的初始值开始进行优化能确定全局最小，或验证只有一个最小值。在可能的情况下可使用不同的方法来验证结果。

问题2：函数 fminu 给出警告信息，并且在极值附近表现出较慢的收敛速度。

推荐：若没有提供梯度，而中断判据又是严格的，函数 fminu 在极值点附近常常会因为在梯度计算中的截断误差而收敛较慢。放松中断条件，则可以加快收敛，而解的精度则会降低。另外运用 options(16:17)改变有限微分干扰水平，可能会提高梯度计算的精度。

问题3：对得到的极值点不能计算函数 f 或 g 的值。

推荐：当不能计算函数值时，可对自变量加上范围条件或者使用罚函数对 f 和 g 给一个较大的值。对梯度计算而言，罚函数必须是光滑连续的。

问题4：极值计算的函数不连续。

推荐：要根据连续函数的一阶和二阶导数来选取求极值的基本方法。当极值点不在间断点附近，或者微分参数被调整以跳过不连续的点时，对某些类型的不连续情况也能得到正确的计算结果。参数 options(16)和 options(17)控制有限微分梯度计算中变量 x 的扰动水平。扰动项 Δx 总是在如下的范围之内：

options(16)<Δx<options(17)

问题5：显示警告信息。

推荐：当中断判据过于严格或者极值问题对于变量的变化特别敏感时可能会发生此种情况。它通常会指出有限微分梯度计算中的截断误差或者多项式内插运算中的问题。这些问题通常可能忽略，然而，这种情况下的收敛将会比正常情况下花费更多的时间。

8.4　参数说明向量 options

参数说明向量 options 共有 18 个元素，包含了优化程序中需要用到的参数。若在某优化程序的第一次调用中，向量 options 为空向量，则会自动产生并使用一组缺省的参数；若向量 options 的元素为零，则这些元素也被赋值为缺省值；若 options 被定义但其元素个数小于 18，则其他的元素将会被赋值为缺省值。

向量options中的所有说明参数如表8-3所示。

表8-3

序号	功　能	缺省值	功能说明
1	显示	0	在优化周期中控制输出的数目,0不显示输出,1按表格输出结果,—1隐藏警告信息。
2	对x终止	1e—4	自变量x的最低层精度的终止判据,当所有的终止判据都满足时,优化将会终止。
3	对f终止	1e—4	在求解时对目标函数f所要求的精度的终止判据。
4	对g终止	1e—7	由函数attgoal,constr,minimax和seminf使用的终止判据,也就是在最坏情况下限定条件精度的量度。
5	主要算法	0	选择主要优化算法。
6	SD算法	0	选择搜索方向算法。
7	搜索算法	0	选择搜索方向算法。
8	函数		求最后极值的函数值,对函数attgoal和minimax而言,它包含一个到达因子。
9	梯度检查	0	当置为1时,在最初的几个迭代周期,梯度将与由有限微分计算的结果比较,此时梯度函数必须存在。
10	函数计数		函数计算计数器
11	梯度计数		函数梯度计算或有限微分梯度计算或有限微分梯度计算的总数。
12	限定计数		限定梯度计算或有限微分梯度计算或有限微分梯度计算的总数。
13	等式约束条件	0	等式限定条件的数目,等式限定条件必须放置在变量g的前几个元素中。
14	最大迭代次数	100n	迭代的最大次数,此值被置为n的100倍,其中n为自变量的数目。在函数fmins中,缺省值为200n,在函数fmin中此缺省值为500n。
15	使用的目标	0	尽可能接近goals的目标数,由函数attgoal使用。
16	最小摄动	1e—8	在有限微分梯度计算中变量的最小变化,对梯度计算而言,实际使用的摄动将会自动调整以提高精度,它将在最小摄动和最大摄动之间变化。
17	最大摄动	0.1	在有限微分梯度计算中变量的最大变化。
18	步长		步长参数,一般说来在第一准迭代中它被保守地赋值为1或更小,这取决于导数的情况。

习 题 八

1. 试求 $f(x)=e^{-1.2x_1}(5x_2^3+4x_2^2+2x_1x_2-x_2+3)$ 的无条件极小值。要求给出极小值结果、迭代计算次数。

2. 试求 $f(x)=100(x_2-x_1^2)^2+(1-x_1)^2$ 的无约束极值点。

3. 试判断条件约束在 $X=[1.002,4.899]^T$ 点是否为

$$\min F(x)=\min\{4x_1-x_2^3-12\}$$

$$g_1(x)=25-x_1^2-x_2^2=0$$

$$g_2(x)=10x_1-x_2^2+10x_2-x_2^2-34\geqslant 0$$

的极值点。

4. 已知目标函数

$$F(x)=4(x_1-5)^2+(x_2-6)^2$$

其约束条件

$$g_1(x)=x_1^2+x_2^2-64\geqslant 0$$

$$g_2(x)=x_2-x_1-10\leqslant 0$$

$$g_3(x)=x_1-10\leqslant 0$$

试求极小值。

5. 已知目标函数 $F(x)=\frac{3}{2}x_1^2+\frac{1}{2}x_2^2-x_1x_2-2x_1$。试求起点为 $x=[2,2]$ 的极小值。

6. 某一薄板空心钢梁结构优化设计时，建立的目标函数为 $F(x)=120x_1+x_2$，其约束条件为：

$$g_1(x)=x_1\geqslant 0$$

$$g_2(x)=x_2\geqslant 0$$

$$g_3(x)=0.25x_2-1\geqslant 0$$

$$g_4(x)=\frac{7}{45}x_1x_2-1\geqslant 0$$

$$g_5(x)=\frac{7}{45}x_1x_2-1\geqslant 0$$

$$g_6(x)=\frac{1}{321}x_1x_2^2-1\geqslant 0$$

试求目标函数的极小值。

7. 一级圆柱齿轮减速器如图 1 所示。设输入功率为 280kw，输入转速为 1 000r/min，传动比 $i=5$，齿轮接触强度许用应力 $[\sigma_H]=8\,550\ \mathrm{N/cm^2}$，齿轮弯曲许

用应力$[\sigma_{b1}]=26\ 100\ \mathrm{N/cm^2}$和$[\sigma_{b2}]=21\ 300\ \mathrm{N/cm^2}$，轴的允许挠度$[f]=0.003\ 1$，轴的许用弯曲应力$[\sigma_u]=5\ 500\ \mathrm{N/cm^2}$。保证齿轮减速器承载能力的条件下，重量按体积计算最轻的目标函数为

$$f(x)=0.785539815(4.75x_1x_2^2x_3^2+85x_1x_3^2-85x_1x_3^2+0.92x_1x_6^2$$
$$-x_1x_5^2+0.8x_1x_2x_3x_6-1.6x_1x_3x_6+x_4x_5+x_4x_6^2+28x_5^2+32x_6^2)$$

约束条件为：

$g_1(x)=x_2-17\geqslant 0$

$g_2(x)=(x_1/x_3)-16\geqslant 0$

$g_3(x)=35-(x_1/x_3)\geqslant 0$

$g_4(x)=x_3\geqslant 0$

$g_5(x)=30-x_3x_3\geqslant 0$

$g_6(x)=x_6-13\geqslant 0$

$g_7(x)=x_5-100$

$g_8(x)=15-x_5\geqslant 0$

$g_9(x)=20-x_6\geqslant 0$

$g_{10}(x)=x_4-x_1-0.5x_6-4\geqslant 0$

$g_{11}(x)=8550-441670/(x_2x_3x_1^{1/2})\geqslant 0$

$g_{12}(x)=2610-\{70980/[x_1x_2x_3^2(0.169+66**10^{-4}x_2-88*10^{-6}x_2^2)]\}\geqslant 0$

$g_{13}(x)=2130-\{70980/[x_1x_2x_3^2(0.2824+17*10^{-4}x_2-39.4*10^{-6}x_2^2)]\}\geqslant 0$

$g_{14}(x)=0.003x_4-0.01233x_4^3/(x_2x_3x_5^4)\geqslant 0$

$g_{15}(x)=550-\{[21085*10^4x_4^2/(x_2^2x_3^2)+2507*10^4]^{1/2}/x_5^3\}\geqslant 0$

$g_{16}(x)=550-\{[21085*10^4x_4^2/(x_2^2x_3^2)+6267*10^4]^{1/2}/x_6^3\}\geqslant 0$

其变量为

$$X=\begin{bmatrix}x_1\\x_2\\x_3\\x_4\\x_5\\x_6\end{bmatrix}=\begin{bmatrix}b\\z_1\\m\\l\\d_1\\d_2\end{bmatrix}$$

8. 齿数z_1为整型变量，模数m为离散型变量(应符合国家标准值)，试求其优化方案。(答案：最优方案为$b=13.0929(\mathrm{cm})$，$z_1=18.7388$，$m=0.8183(\mathrm{cm})$，$d_1=10.0001(\mathrm{cm})$，$d_2=13.0000(\mathrm{cm})$，$l=23.5930(\mathrm{cm})$，体积$f(x)=35334.3583$

(cm^3)。凑整方案为 $b=13.0929$(cm)，$z_1=19$，$m=0.9$(cm)，$d_1=10.0001$(cm)，$d_2=13.0000$(cm)，$l=23.5930$(cm)，$f(x)=40709.3752$(cm^3))。

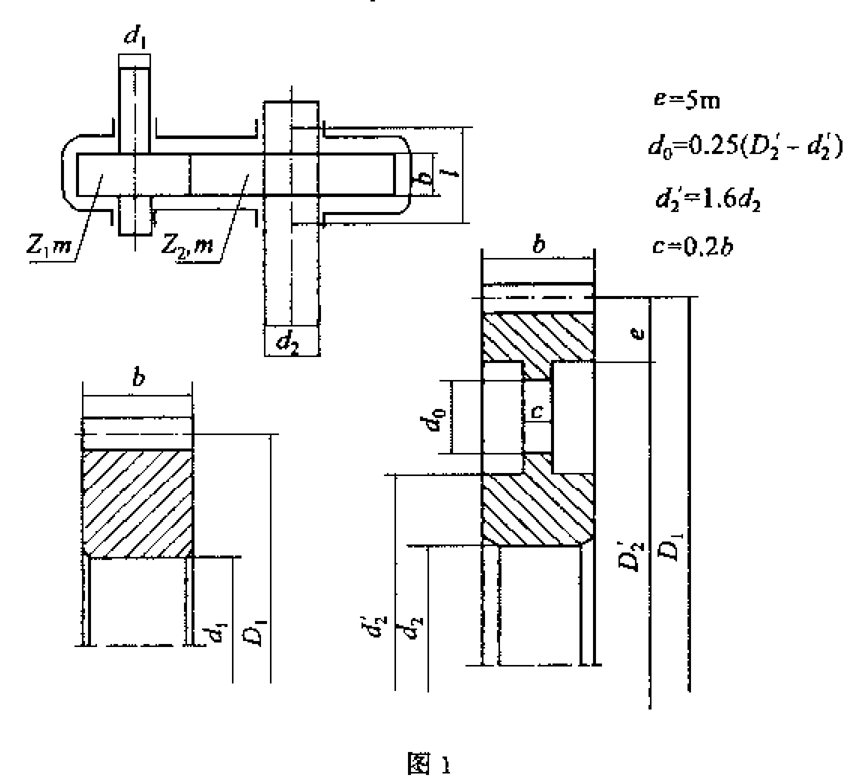


图 1

第九章 自动控制设计方法

当今自动控制技术在人类生产活动中，起着越来越重要的作用，指导自动控制系统的分析和设计的控制理论也有了很大发展。早期的经典控制理论主要应用于单输入单输出、定常系统的分析和设计，分析及设计方法较为简单。现代控制理论具有更广泛的适应性，可用于多输入多输出、定常或时变系统的分析和设计，所讨论的问题更加复杂，涉及的领域更为广泛。由于科学技术和生产的迅速发展，对系统实行控制的要求不断提高，控制算法越来越复杂，控制器的设计更加困难，传统的分析及设计方法已很难满足设计要求，随着计算机技术的发展，控制系统的计算机辅助设计技术得到广泛应用，利用计算机对控制系统进行建模、分析、设计及仿真，减轻了繁杂的设计工作，提高了设计质量。由于MATLAB强大的数值计算功能，使其在控制系统的计算机辅助设计中得到广泛应用。

9.1 MATLAB关于控制系统模型命令

控制系统的动态行为常用三种数学模型描述：1. 系统的传递函数；2. 根轨迹方程；3. 状态空间表达式。每种模型均有连续和离散的形式，它们各有特点，适于不同应用场合。为方便解决问题，有时需在不同模型之间进行转换。MATLAB处理矩阵对象，下面介绍如何用矩阵表示不同的控制系统模型。

9.1.1 连续系统

1. 传递函数模型

单输入单输出控制系统的传递函数

$$G(s)=\frac{b_m s^m+b_{m-1}s^{m-1}+\cdots+b_0}{a_n s^n+a_{n-1}s^{n-1}+\cdots+a_0} \qquad (9.1.1\text{-}1)$$

对于线性时不变系统，上式中的a_i,b_i都是常数，且$a_n\neq0$。在MATLAB中，可用分子、分母的系数构成的向量表示：

$num=[b_m\quad b_{m-1}\quad \cdots\quad b_0]$，$den=[a_n\quad a_{n-1}\quad \cdots\quad a_0]$.

注意这里分子、分母向量的元素按传递函数分子、分母的降幂排列。

2. 根轨迹方程

古典控制论中，当用根轨迹法对线性系统进行设计时，需将传递函数变换为根

轨迹方程：

$$G(s)=k\frac{(s-z_1)(s-z_2)\cdots(s-z_m)}{(s-p_1)(s-p_2)\cdots(s-p_n)}=\frac{k\prod_{i=1}^{m}(s-z_i)}{\prod_{j=1}^{n}(s-p_j)} \tag{9.1.1-2}$$

z_i，p_j 是系统的零、极点，k 是系统根轨迹增益。可以用[z，p，k]向量组表示它们，其中

$z=[z_1 \quad z_2 \quad \cdots \quad z_m]$， $p=[p_1 \quad p_2 \quad \cdots \quad p_n]$.

对于单输入单输出系统，k 是标量。

3. 状态空间表达式

用状态空间表达式描述系统的动态过程：

$$\begin{cases}\dot{x}=Ax+Bu\\ y=Cx+Du\end{cases} \tag{9.1.1-3}$$

在 MATLAB 中，可以直接用(A，B，C，D)表示，A，B，C，D 即状态空间表达式的系数矩阵，各矩阵都被看做独立变量。

【例 9.1.1-1】 显示传递函数。

$$\frac{s+1}{s^3+3s^2+2s}$$

```
num=[1 1]; den=[1 3 2 0];
% num 为分子多项式系数，den 为分母多项式系数
  printsys(num,den)
```

【例 9.1.1-2】 显示以状态方程表示的系统。

```
A=[-.0558 -.9968 .0802 .0415; .598 -.115 -.03180; -3.05 .388 -.4650; 0 0.0805 1 0];
B=[.0729 .0001; -4.75 1.23; 1.53 10.63; 0 0];
C=[0 1 0 0; 0 0 0 1];
D=[0 0; 0 0];
states={'侧滑角' '输出方向舵偏角' '横滚角速度' '内倾角'};
inputs={'输入方向舵偏角' '副翼偏角'};
outputs={'航向角速度' '侧滑内倾角'};
sys=ss(A,B,C,D,...
'statename',states,'inputname',inputs,'outputname',outputs)
```

连续函数三种表达形式可相互转化，参见 9.1.4 节。

9.1.2 离散系统

1. 传递函数模型

离散系统的动态特性可用差分方程或脉冲传递函数表示。脉冲传递函数是输出信号与输入信号的 z 变换之比：

$$H(z)=\frac{b_m z^m+b_{m-1}z^{m-1}+\cdots+b_0}{a_n z^n+a_{n-1}z^{n-1}+\cdots+a_0},\quad n\geqslant m \tag{9.1.2-1}$$

2. 零极点模型

在研究系统结构和参数对其动态性能的影响时，可以将系统的脉冲传递函数表示为零极点的形式：

$$H(z)=k\,\frac{(z-z_1)(z-z_2)\cdots(z-z_m)}{(z-p_1)(z-p_2)\cdots(z-p_n)} \tag{9.1.2-2}$$

3. 状态空间模型

离散系统的状态空间表达式：

$$x(k+1)=ax(k)+bu(k)$$

$$y(k+1)=cx(k+1)+du(k+1) \tag{9.1.2-3}$$

4. 连续系统与离散系统的互相转化

在设计控制系统时，为方便分析、求解问题，需要在连续系统与离散系统模型间进行转换。MATLAB 控制系统工具箱有一组可将连续系统与离散系统互相转化的函数。

(1)将连续系统变为离散系统

Dsys=c2d(Csys,Ts)

Dsys=c2d(Csys,Ts,method)

语句 c2d(Csys,Ts) 离散连续时间线性时不变系统 Csys，输入信号加零阶保持器，Ts 是采样时间。c2d(Csys,Ts,method)中字符串 method 表示转换方法，method 可以是以下几种：

(a)'zoh' 对输入信号加零阶保持器；

(b)'foh' 对输入信号加一阶保持器；

(c)'tustin' 双线性变换(图斯汀算法)；

(d)'prewarp' 频率转折图斯汀算法；

(e)'matched' 零极点匹配法。

c2d 还可用于有输入延迟的 LTI 系统(用方法'zoh' 和'foh')及 MIMO 系统(零极点匹配法 matched 除外)。

【例 9.1.2-1】 有一系统的传递函数是

$G(s)=\frac{10}{s^2+10s}$，设采样周期 0.1 秒，试将其离散化。

输入：

```
H=tf([10],[1 10 0])
```

```
Hd=c2d(H,0.1)
```

命令窗口显示：

```
Transfer function:
 0.03679z+0.02642
------------------
z^2-1.368z+0.3679
Sampling time: 0.1
```

(2)将离散系统转变为连续系统

```
Csys=d2c(Dsys)
Csys=d2c(Dsys,method)
```

字符串'method'意义见 c2d( )。

【例 9.1.2-2】 有一系统的脉冲传递函数是

$G(s)=\frac{z-1}{z^2+z+0.3}$，设采样周期 0.1 秒，试将其转化为连续系统传递函数。

输入：

```
H=tf([1 -1],[1 1 0.3],0.1);
Hc=d2c(H);
```

命令窗口显示：

```
121.7s+1.878e-0.12
------------------
s^2+12.04s+776.7
Transfer function:
Sampling time:0.1
```

9.1.3 系统模型建立

实际的控制系统往往由若干环节组成，各环节有各自的数学模型。建立系统模型时要将各环节综合起来，综合的方法主要有并联、串联和反馈。常用的系统结构图中的变换命令：

1. 串联：$G_s(s)=G_1(s)G_2(s)$

```
sys=series(sys1,sys2)
```

【例 9.1.3-1】 已知两传递函数，求其串联连接的等效传递函数。

输入：

```
num1=3;
den1=[1 4];
num2=[2 4];
den2=[1 2 3];
[num den]=series(num1,den1,num2,den2)
```

命令窗口显示：

```
num=0    0    6    12
den=1    6    11   12
```

2. 并联：$G_p(s)=G_1(s)+G_2(s)$

sys=parallel(sys1,sys2)

【例 9.1.3-2】 已知两传递函数，求其并联连接的等效传递函数。

输入：

```
num1=3;
den1=[1 4];
num2=[2 4];
den2=[1 2 3];
[num den]=parallel(num1,den1,num2,den2)
```

命令窗口显示：

```
num=0    5    18    25
den=1    6    11    12
```

3. 反馈：$G_f(s)=\frac{G(s)}{1+G_1(s)G_2(s)}$

sys=feedback(sys1,sys2,sign)

sign 的值为+1 或−1，表示反馈性质；+1 表示正反馈，−1 表示负反馈，缺省为负反馈。

【例 9.1.3-3】 已知两传递函数，求其反馈连接的等效传递函数。

输入：

```
num1=3;
den1=[1 4];
num2=[2 4];
den2=[1 2 3];
[num den]=feedback(num1,den1,num2,den2)
```

命令窗口显示：

```
num—0    3    6    9
den=1    6    17   24
```

单位反馈：$G_c(s)=\frac{G(s)}{1+G(s)}$

G_c=feedback(G,1,sign)

【例 9.1.3-4】 已知传递函数，求其单位反馈连接传递函数。

输入：

```
num1=3;
den1=[1,4];
```

```
[num,den]=feedback(num1,den1,1,1)
```

命令窗口显示：

```
num=0    3
den=1    7
```

9.1.4 模型之间转换

同一个系统可用三种不同的模型表示。为分析系统，需要在三种模型之间进行转换。MATLAB 提供了有关函数，函数功能说明如下：

1. [z,p,k]=tf2zp(num,den)

传递函数转换为零极点模型。对于 SISO 系统，传递函数 z,p,k 分别表示零点列向量、极点列向量和增益常量。

2. [num,den] = zp2tf(z ,p,k)

零极点模型转换为传递函数。

【例 9.1.4-1】 求零极点与增益 k。

输入：

```
num=[1 -0.5  2];
den=[1 0.4  1 ];
[z,p,k]=tf2zp(num,den) %零点 z,极点 p,增益 k
```

命令窗口显示：

```
z=0.2500 + 1.3919i 0.2500 -1.3919i
p=-0.2000 + 0.9798i  -0.2000 - 0.9798i
k=1
```

3. [A,B,C,D]=tf2ss(num,den)

单输入系统传递函数转换成状态空间表达式。向量 den 表示传递函数分母多项式的系数(按 s 降幂排列)，num 是矩阵，每行代表一个输出的多项式系数，有多少输出，就有多少行。A,B,C,D 是可控标准形的系数矩阵。

4. [num,den]=ss2tf(A,B,C,D,iu)

A,B,C,D 是系统系数矩阵，iu 是系统输入编号。num 是传递函数分子多项式系数组成的矩阵，每行对应一个输出。

5. [Z,p,k]=ss2zp(A,B,C,D,iu)

函数参数意义同 4。Z 是矩阵，其列数与输出 y 的维数一样，每列对应一个输出的零点，列向量 k 是分子传递函数的增益。

6. [A,B,C,D]=zp2ss(Z,p,k)

SIMO 系统的零极点模型转换成状态空间表达式。p 是极点向量，Z 是矩阵，其列数等于输出的个数，每列对应一个输出，k 是增益向量。

9.1.5 MATLAB 其他有关系统模型的函数

1. 模型降阶与实现

(1)rsys=modred(sys,elim)

消去向量 elim 指定的状态,获得降阶模型。常与 balreal 连用。

(2)sysb=balreal(sys)

[sysb,g,T,Ti]=balreal(sys)

求系统的状态空间均衡实现。向量 g 是格拉姆矩阵对角线元素,T,Ti 是变换阵及其逆。

(3)sysr=minreal(sys)

sysr=minreal(sys,tol)

消去状态空间模型中的不可控、不可观子系统;传递函数零极点对消。tol 是容差,缺省值 tol=sqrt(eps)。当零极点非常接近时,可通过改变容差的大小,强制对消。

2. 模型性质

(1)[Abar,Bbar,Cbar,T,k]=ctrbf(A,B,C)

[Abar,Bbar,Cbar,T,k]=ctrbf(A,B,C,tol)

按可控性分解系统。T 为变换矩阵,向量 K 元素个数等于系统阶数,sum(k)表示可控子空间的状态变量数。tol 为容差,缺省值 10*n*norm(A,1)*eps。

(2)[Abar,Bbar,Cbar,T,k]=obsvf(A,B,C)

[Abar,Bbar,Cbar,T,k]=obsvf(A,B,C,tol)

按可观性分解系统。各参数意义同上。

(3)Ctr=ctrb(A,B)

求可控性矩阵。

(4)Ob=obsv(A,C)

求可观性矩阵。

(5)csys=canon(sys,'type')

[csys,T]=canon(sys,'type')

求系统矩阵的标准型。字串 type 值是 model 时,csys 的系统矩阵是约当标准型;是 companion 时,csys 是可控、可观标准Ⅱ形的系统矩阵。T 是转换矩阵。

9.1.6 稳定性分析

对连续系统,如果系统的所有极点都位于 s 平面左半部,即 Re(pi)<0,i=1,2,… ,则系统是稳定的。因此可以根据极点的值判断系统是否稳定。下面是求零极

点函数。

1. p=pole(sys)

求系统的极点。

2. z=tzero(sys)

[z,gain]=tzero(sys)

求系统的零点、gain SISO 的增益。

3. [p,z]=pzmap(num,den)

求极点、零点及绘制图形。如果该命令中没有输出变量，执行该命令后，绘制系统零极点图。

【例 9.1.6-1】 已知 H=tf([2 5 1],[1 2 3])，求零极点，并画零极点图。

输入：

```
[p,z]=pzmap(H)
pzmap(H)
```

第二行画零极点图，图中'o' 表示零点，'*'表示极点。

9.2 时 域 分 析

对控制系统来说，系统的数学模型实际上是某种微分方程或差分方程模型。由于 MATLAB 长于数值运算，因而可以在MATLAB 中仿真系统的过渡过程。根据系统的运动方程，以某种数值算法从给定的初始值出发，逐步计算出每一时刻系统的响应(即系统的时间响应)，并绘出系统的响应曲线，由此来分析系统的性能。时间响应反映了系统对输入和扰动的瞬态行为，系统的稳态性、暂态性、稳态精度，系统响应的上升时间、调节时间、超调和稳态误差都能从时间响应上反映出来。控制系统工具箱提供了对系统阶跃响应、脉冲响应等进行仿真的函数。

1. [y,t,x]=step(num,den,t)

计算线性时不变系统(continuous,discrete,SISO,MIMO)的阶跃响应。无左值调用时，绘制系统的阶跃响应。t 为响应时间，可以是单值(表示终了时间)或是均匀分布的向量。对于单输入多输出 SIMO 系统，输出结果是一个矩阵，该矩阵的列与输出数量相同。

【例 9.2-1】 计算并绘制传递函数 $G(s)=\frac{10}{s^2+2s+10}$ 的阶跃响应($t=0$ 至 $t=10$)

输入：

```
num=10;den=[1 2 10];
```

```
t=[0:0.1:10];%或者t=10
step(num,den,t);%绘制系统的阶跃响应
```

2. [y,t,x]=impulse(num,den,t)

计算线性时不变系统的脉冲响应。无左值调用时，绘制系统的脉冲响应。其他参数意义同1。

3. y=lsim(num,den,u,t)

计算线性时不变系统对任意输入的响应。无左值调用时，绘制系统的响应图形。t为均匀间隔的时间向量，u为输入矩阵，行数等于t的维数，列数等于输入；每一行u(i,:)表示时间t(i)的各输入值。

【例9.2-2】 模拟系统的斜坡响应。

输入：

```
ramp=t;lsim(num,den,ramp,t)   %t,num,den 同上
```

【例9.2-3】 计算10s的噪声响应。

```
noise=rand(101,1)
y=lsim(num,den,noise,t)
```

9.3 根轨迹法

控制系统闭环传递函数的极点决定了系统的稳定性和动态响应的基本特性，系统的闭环零极点在S面的分布影响系统的动态性能。因此用古典方法分析、设计、控制系统时，都要研究系统闭环零极点在S面的分布，以及系统参数变化对它们的影响。闭环极点可通过解系统闭环特征方程求出。控制系统的闭环特征方程很多是高次代数方程，虽然可用数值方法求解，但不能看出系统某些参数变化对零极点在S面的分布的影响。1949年伊文斯(W. R. Evans)提出了由开环传递函数确定闭环特征根的图解法，即根轨迹法。根轨迹是开环系统某一系统参数从零变到无穷时，闭环系统特征方程式根在S平面上的轨迹。一般选开环系统的增益k为变化参数。

通常绘制系统的根轨迹图是很麻烦的，因此在教科书中经常采用一些简单系统的图例。现在有了MATLAB等软件的支持，绘制根轨迹图就变得十分简单了。

1. [r,k]=rlocus(num,den)

r=rlocus(num,den,k)

计算系统的根轨迹。无左值时，输出根轨迹图形。向量k是所有增益值，r是闭环极点复数矩阵，有k列，第j列即所有对应k(j)的闭环极点。如果指定k，命令可按照给定的参数绘制根轨迹图，否则增益是自动确定的。

【例 9.3-1】 绘制系统的根轨迹。

$$H(s)=\frac{k}{s(0.5s+1)}$$

输入：

```
num=[1];
den=[ 0.5 1 0 ];
rlocus(num,den )
```

2. sgrid

sgrid(z,wn)

在 S 面上以阻尼系数和自然频率为坐标画栅格，参数 z,wn 指定阻尼系数与自然频率的范围，如不指定，阻尼从 0 到 1，步长为 0.1；自然频率从 0 到 10，步长为 1red/sec。

【例 9.3-2】

输入：

```
num=[2 5 1];den=[1 2 3];axis([-1 1 0 3]);
rlocus(num,den)
sgrid([0.5;0.1;0.7],0.5   %连续系统的 ωn-ξ 网格根轨迹图
```

3. [k,poles]=rlocfind(num,den)

确定根轨迹上某一点的增益。执行命令后，将在图形屏幕上生成一个十字光标，移到光标所希望的位置，单击左键，将得到该极点的位置及对应的增益 k 值。如果所选择的点接近于根轨迹上某点，则该点对应的增益值及极点位置将作为命令的输出参数。该命令也可以在没有绘制根轨迹图之前执行，此时使用如下命令：

[k,poles]=rlocfind(num,den,p)

9.4 频域分析

系统的频率响应是经典控制论中用以描述系统性能的主要方法之一，它从系统的频率特性出发分析系统，研究不同频率下系统的行为。其基本原理是：若一个线性系统受到频率为 ω 的正弦信号激励，其输出幅值与输入信号成 $M(\omega)$ 比例关系，输出与输入之间有相位差 $\Phi(\omega)$，$M(\omega)$ 和 $\Phi(\omega)$ 是关于 ω 的有理函数，可以通过 $M(\omega)$ 和 $\Phi(\omega)$ 来表示系统的特征。频率响应研究系统的频率行为，从频率响应中可得带宽、增益、转折频率、闭环稳定性等系统特征。若知道系统的这些特征，可防止结构谐振、抑制噪声，改善系统的性能。控制系统工具箱提供了绘制波德(bode)图、尼柯尔斯(nichols)图的函数以及判别系统稳定性的奈奎斯特(nyquist)判据函数，下面介绍这些函数。

9.4.1 绘制对数频率特性图(波德bode图)

```
bode(sys)
bode(sys,w)
[mag,phase,w]=bode(sys)
```

计算系统的频率响应的幅值、相位,无左值时,绘制幅相曲线。参数w表示要求的频率范围。

```
bode(a,b,c,d)        %ẋ=ax+bu y=bx+du 的每个 bode 图
bode(a,b,c,d,iu)     %第 iu 个输入的 bode 图
bode(num,den)
```

在同一屏幕中的上下两部分分别生成波德幅值图(以db为单位)和波德相平面图(以rad为单位)。在另外的格式中,返回的幅值与相角值为列向量(幅值不是以db为单位)。

```
[mag,phase,w]=bode(num,den)
```

计算系统的频率响应及对应频率向量w。

```
[mag,phase]=bode(num,den,w)
```

【例9.4.1-1】 自然频率 $\omega_n=1$,阻尼因子 $\xi=0.2$。

```
[a,b,c,d]=ord2(1,0.2)
bode(a,b,c,d)
title('BODE PLOT')
```

9.4.2 绘制幅相特性曲线的极坐标图(奈奎斯特nyquist图)

```
[re,im]=nyquist(num,den,w)
```

nyquist命令可计算G(jω)的实部和虚部,在复平面上绘制虚部与实部的轨迹,即nyquist图。

【例9.4.2-1】

```
clf
w=[0:0.01:5]
num=[2 5 1]
den=[1 2 3]
%nyquist(num ,den)
nyquist(num ,den,w)
title('Nyquist Plot')
```

9.4.3 绘制对数幅相特性图(尼柯尔斯nichols图)

```
[mag,im]=nichols(num,den,w)
```

【例 9.4.3-1】

```
w=[0:0.01:5]
num=[2 5 1]
den=[1 2 3]
% nichols(num ,den)
nichols(num ,den,w)
title('Nichols Plot')
```

nichols 函数计算系统的频响幅值与相角(以角度为单位)。有左值时，nichols 与 bode 的输出结果一样。

9.4.4　控制系统相对稳定性参数(增益裕量与相角裕量)

[gm,pm,wpc,wgc]=margin(mag,phase,w)

输入参数为幅值 mag(不是以 dB 为单位)、相角 phase 和频率向量 w(各参数可由 bode 或 nichols 得到)。

输出参数是增益裕量(不是以 dB 为单位)、相角裕量(以角度为单位)和它们所对应的频率。

如果命令格式没有左值，绘制带有裕量标记的(垂直线)波德图。如果在轴上有多个穿越频率，图中则标出稳定裕量最坏的那个标记。

```
[a,b,c,d]=ord2(1,0.2)    %ord2(Wn,Z) 产生二阶系统 A,B,C,D
[mag,phase,w]=bode(a,b,c,d,1);
margin(mag,phase,w)
```

9.5　极点配置和状态估计器

控制系统极点在 S 面上的分布，表征系统的性能。作为综合系统性能指标的一种形式，在设计控制系统时，往往给定一组期望极点，或根据设计指标求出相应的极点。根据这组极点设计的系统，其动态性能可以满足控制系统的设计要求。根轨迹法通过改变系统某一参数，使闭环系统的极点沿其轨迹移动，改变极点在 S 面的位置。现代控制论则是通过选择反馈增益矩阵，把闭环系统的极点配置在所希望的位置，使系统满足设计要求。MATLAB 用状态空间法实现控制系统的极点配置。我们知道，利用状态反馈可以达到镇定、解耦或最优化控制系统的目的。状态反馈假设所有状态变量都是可以检测的，否则无法实现状态反馈控制率。但这点在实际应用中很难做到，因此提出了状态估计器。状态估计器的作用就是用可检测的输入和输出(状态估计器存在的条件是系统可观或不可观子系统渐进稳定)，产生充分

逼近状态变量的输出，用状态估计器的输出代替状态变量，满足状态反馈的要求。本节讨论有关极点配置和状态估计器的函数。

9.5.1 极点配置

MATLAB 用状态空间法进行极点配置，如果系统不是状态空间表达式，先用 ss 函数进行转换。

1. K=place(A,B,p)

[K,prec,message]=place(A,B,p)

给定单输入或多输入系统 $\dot{x}=Ax+Bu$ 和期望极点向量 p，求状态反馈增益矩阵 K。p 的维数与 A 的行数一样。prec 表示 A−BK 的特征值逼近 p 的程度，如果非零极点偏离期望值超过 10%，message 返回警告信息。注意反馈控制率是 $u=-Kx$。

【例 9.5.1-1】 有系统状态方程为

$$\dot{x}=\begin{bmatrix}0 & 1 & 0\\0 & 0 & 1\\0 & -2 & -3\end{bmatrix}x+\begin{bmatrix}0\\0\\1\end{bmatrix}u$$

系统可控且可观。设计状态反馈控制器，使闭环系统的极点为−2，−1，±i1。

输入：

```
a=[0 1 0; 0 0 1; 0 -2 -3]
b=[0; 0; 1]
p=[-2 -1+1i -1 -1i]
k=place(a,b,p)
```

输出：

```
k=4.0000     4.0000     1.0000
```

2. k=acker(A,b,p)

求状态反馈增益矩阵 k，A，b，p 意义同 place。与 place 不同的是，acker 的对象是 SISO 系统。一般推荐用 place，即使是 SISO 系统。

9.5.2 状态估计器

如果系统不可观，就无法实现反馈控制率 $u=-Kx$。状态估计器的作用就是产生一个充分逼近状态变量 x 的向量，用它去代替 x，实现系统的可观性。

1. est=estim(sys,L)

给定状态空间模型 sys，估计器增益 L，返回估计器的状态方程 est。

对于状态空间表达式 $\begin{cases}\dot{x}=Ax+Bu\\ y=Cx+Du\\ \dot{\hat{x}}=A\hat{x}+L(y-C\hat{x})\end{cases}$

estim 根据估计器表达式$\begin{bmatrix}\hat{y}\\ \hat{x}\end{bmatrix}=\begin{bmatrix}C\\ I\end{bmatrix}\hat{x}$ 计算系统输出和状态的估计 $\hat{y}$,$\hat{x}$。

增益矩阵 L 可用 place 求得。

9.6 控制系统实例分析

【例 9.6-1】 使用 MATLAB 命令，对下面的系统进行分析。系统传递函数如下：

$$G(s)=\frac{1}{s(s+1)(s+2)^2},\quad K=1.5$$

其结构见图 9.6-1。

频率响应分析：首先写出系统的开环传递函数，得到对应 $K=1.5$ 时的系统频率特性波德图。用 logspace 命令生成具有 100 个频率点的对数坐标轴。

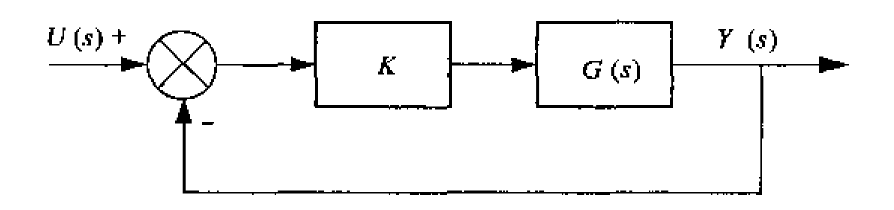


图 9.6-1　系统框图

```
k=1.5;ng=1;dg=poly([0-1-2]);
w=logspace(-1,1,100)';
[m,p]=bode(k*ng,dg,w);
subplot(211),semilogx(w,20*log10(m))
subplot(212),semilogx(w,p)
```

由波德图(图 9.6-2)可见，系统的增益裕量与相角裕量大约为 10dB 与 45°，可以用 margin 命令验证。

```
[gm pm wgc wpc]=margin(m,p,w)
gm=4.002
pm=41.5332
wgc=1.4142
```

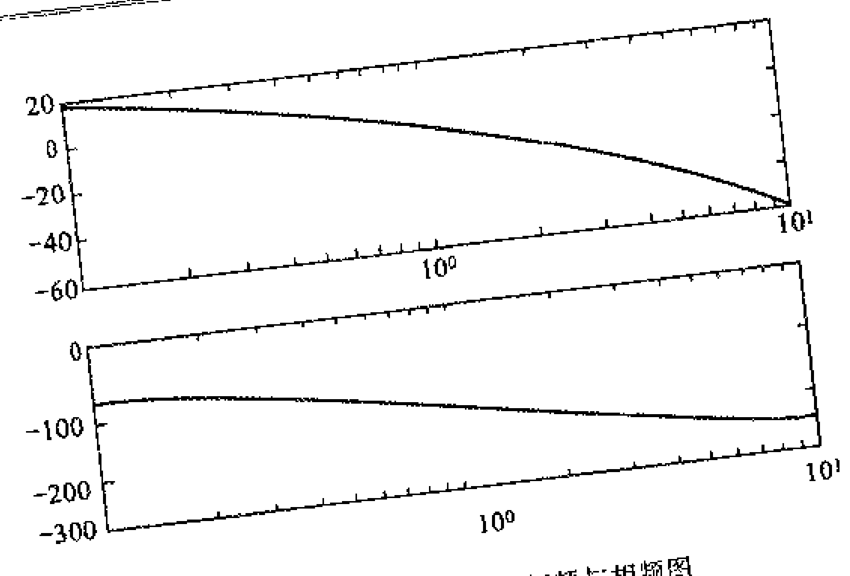


图 9.6-2　系统 KG(s)的幅频与相频图

wpc=0.6118

函数输出 gm,pm,wgc,wpc 分别表示增益裕量、相角裕量及对应的剪切频率。gm 值表示系统增益再增加 gm(= 4),系统将转入不稳定。

下面介绍如何求得带有单位圆的奈奎斯特图。为生成一个单位圆,使用 linspace 命令建立一个有 100 个元素的向量(元素值 0～2π)。这个单位圆是通过在复平面上绘制 e^{jw}的虚部与实部的轨迹而得到的。

```
Clf;
w2=linspace(0,2*pi,100);
ejw=exp(j*w2);
r2=real(ejw);
i2=imag(ejw);
plot(r2,i2)  %画单位圆
axis square
%下面的命令绘制带有单位圆的奈奎斯特图
clf;
[r i]=nyquist(k*ng,dg,w);
plot(r2,i2,r,i)
grid
axis([-1 1 -1 1 ])
axis square
```

可以看到图 9.6-3 中的轨迹没有包围(-1,j0)点,所以它的系统是稳定的。注

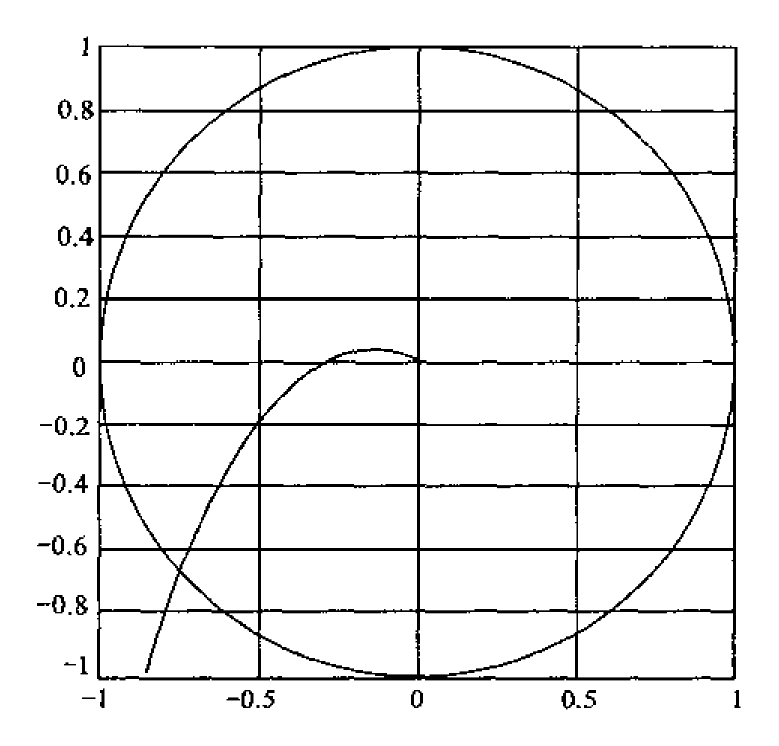


图 9.6-3 【例 9.6-1】系统的奈奎斯特图

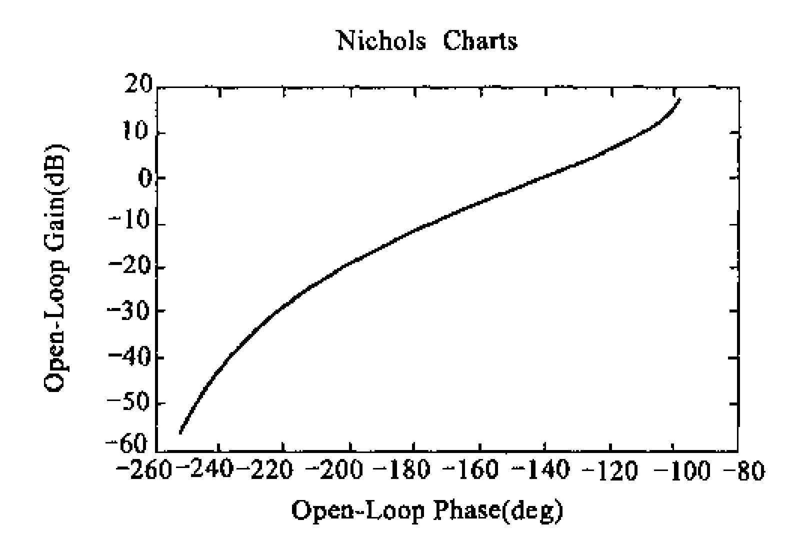


图 9.6-4 KG(s)的尼柯尔斯图

意在图 9.6-4 中的增益裕量是 12dB。

```
clf;
```

```
nichols(k*ng,dg,w)   % 或 plot(p,20*log10(m))
axis([-360 0 -60 20 ])
```

从闭环系统的波德图中，可以得到系统带宽及谐振峰值等特征参数。设 T(s) 为系统闭环传递函数，用 cloop 命令求系统的闭环传递函数：

```
clf;
[nt,dt]=cloop(k*ng,dg);
mc=bode(nt,dt,w);
subplot(211)
semilogx(w,20*log10(mc))
grid
```

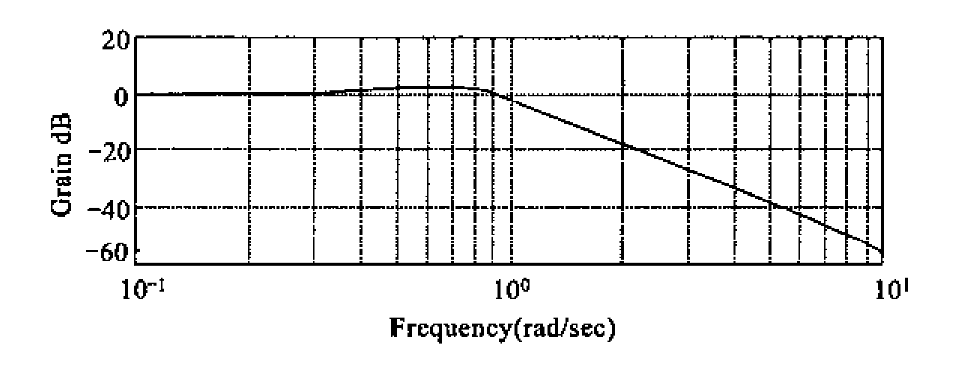


图 9.6-5　闭环系统的波德幅值图

从图 9.6-5 可以看到闭环系统的谐振峰值是 3dB，带宽约为 1rad/s。

根轨迹法分析：下面用根轨迹法分析系统，看增益变化对系统性能的影响。

输入：

```
rlocus(k*ng, dg)
```

得到系统根轨迹图(图 9.6-6)。

使用 rlocfind 命令，可以知道当 K>6 时，系统变得不稳定。还可以通过求得该临界点(K=6)附近一定范围内极点的参数，得到其准确值。

```
clpole=rlocus(ng,dg,[0.3:0.1:7])
rlocus(ng,dg,[0.3:0.1:7])
range=[0.3:0.1:7]'
[range,clpole]
ans=
0.3000   -2.1254   -0.6611           -0.2135
0.4000   -2.1597   -0.4201-0.0932i   -0.4201+0.0932i
5.9000   -2.9909   -0.0046-1.4045i   -0.0046+1.4045i
```

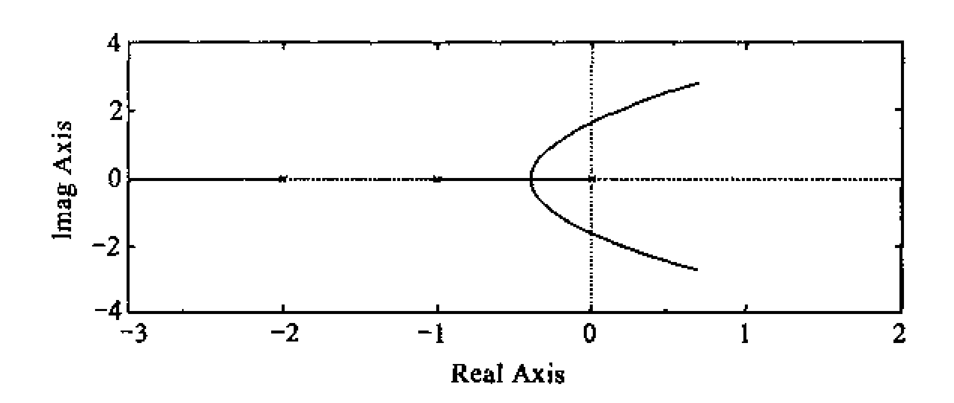


图 9.6.-6　系统根轨迹图

6.0000	−3.0000	0.0000−1.4142i	0.0000+1.4142i
6.1000	−3.0090	0.0045−1.4238i	0.0045+1.4238i

上列数据显示了该区域内增益及对应的闭环极点位置。可以得出如下结论：

(1)0<K<0.4 时，闭环系统具有不同的实数极点，系统处于过阻尼状态。

(2)K=0.4 时，对应为分离点，系统处于临界阻尼状态。

(3)0.4<K<6 时，系统主导极点为共轭复数极点，系统为欠尼状态。

(4)K=6 时，系统有一对虚根，系统处于临界稳定状态。

(5)K>6 时，系统出现右根，系统处于不稳定状态。

为了验证上述结论，我们来求闭环系统的传递函数，并求五个系统的阶跃响应。K 值分别为 0.25，0.4，1.5，6 和 8。下面是求系统阶跃响应的程序。

```
clf
k=1.5,ng=1,dg=poly([0 -1 -2])
rangek=[0.25 0.4 1.5 6 8],t=[0:0.2:20]
for j=1:5
  [ntc,dtc]=cloop(ng*rangek(j),dg)
  y(:,j)=step(ntc,dtc,t)
end
subplot(211),plot (t, y(:,1:3)),grid
subplot(212), plot (t, y(:,4:5)),grid
gtext('k=0.25'),gtext('k=0.4'),gtext('k=1.5'),
gtext ('k=6'),gtext('k=8')
```

图 9.6-7 是这五种系统的阶跃响应(三个是稳定的，两个是不稳定的)。下面讨论欠阻尼(K=1.5)时的情况。

1. 系统阶跃响应的超调量：即峰值与稳态值之差，且与系统稳态值之比。超调

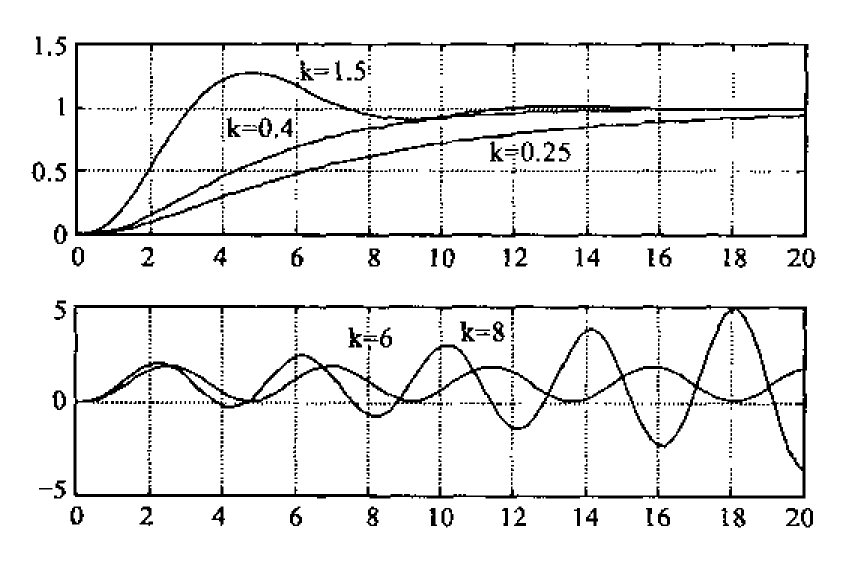


图 9.6-7　不同 K 值的系统阶跃响应

量计算过程如下：

```
y3=y(:,3),mp=max(y3),yss=y3(length(t))
pos=100*(mp-yss)/yss
pos=27.9035
```

2. 稳态误差分析：我们知道，单位反馈系统在零点处有一极点的系统为典型 I 型系统，它的稳态误差 e_{ss} 为零。对于斜坡响应的误差为 $1/K_v$(K_v 为速度误差系数)。

$$k_v=\lim_{s\to 0} S\ G(S)=1.5/2=0.75$$

因此 $e_{ss}=1.33$。

下面使用 lsim 命令去求斜坡输入响应，并且证明所得到的结果(见图 9.6-8)。计算斜坡输入响应的简单办法是在系统闭环传递函数上再乘一个 $1/s$，下面为计算系统斜坡响应的程序。

```
k=1.5,ng=1,dg=poly([0 -1 -2])
[nt, dt]=cloop(k*ng,dg);
dtt=[dt, 0];t2=[0: 0.1: 10];
yramp=step(nt,dtt,t2);
subplot(211),plot(t2,[t2',yramp]),grid
subplot(212), ess=t2'-yramp; plot(t2,ess),grid,
ess(length(ess))
```

由下列命令可以验证其稳态误差：

```
ess=t2-yramp'; subplot(212), plot(t2,ess),
```

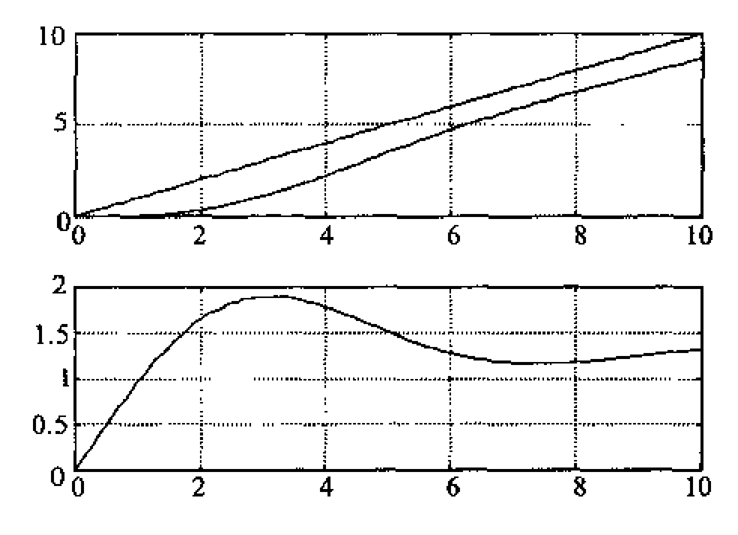


图 9.6-8　系统斜坡响应及误差

```
grid, ess(length(ess))
ans =
  1.3307
```

滤波器性质：由闭环系统的波德图（图 9.6-5）可以看出，系统特性为一低通滤波特性。大多数系统都具有低通滤波特性，这意味着系统对高频信号具有很大的衰减性，可通过随机噪声信号在欠阻尼系统中的时间响应来证明这一点。rand 命令用于将一噪声信号加入到一个阶跃输入信号中，而 lsim 命令用于求解其响应。

```
clf
t=length(t2)
noise=ones(t,1)+rand(t,1);
ynoise=lsim(nt,dt,noise,t2);
  subplot(211),plot(t2,[noise ynoise])
```

上列程序中 t2 是一个向量，ones(t,1)是与 t2 有相同维数、每个元素都是 1 的向量，rand(t,1)建立一个与 t2 有相同维数、元素是 0～1 之间的随机数的向量，两者之和构成了一个噪声阶跃输入信号。为了便于比较，将输入与输出绘制在同一图中（图 9.6-9）。与分析的结果一样，系统对高频噪声信号衰减很大。由于噪声信号具有 0.5 单位的幅值，因此其输出信号中出现了一个 0.5 单位的偏移。通常噪声信号的幅值相对于其输入信号的幅值要小得多，因此其作用几乎可以忽略。

【例 9.6-2】　硬盘读写磁头控制器的设计。

硬盘读写头的运动模型可用下面微分方程表示：

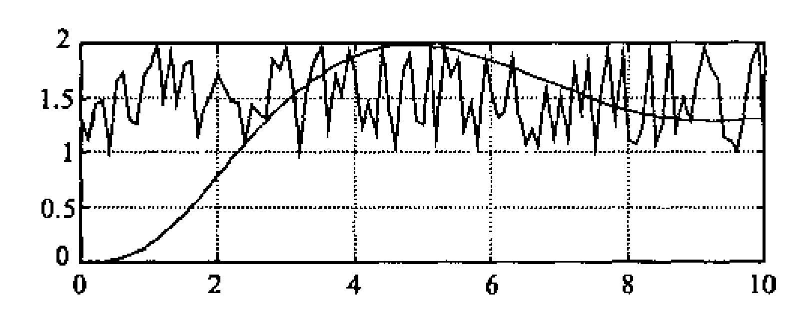


图 9.6-9　随机噪声阶跃输入的系统响应

$$J\ddot{\theta}+C\dot{\theta}+K\theta=K_i i \qquad (9.6\text{-}1)$$

式中 J 是磁头转动惯量；C 是支承的粘滞阻尼系数；K 是回复力常数；Ki 是电机力矩常数；$\ddot{\theta}$,$\dot{\theta}$ 和 θ 分别是角加速度、角速度、角位移；i 是电动机电流。对(9.6-1)式进行拉氏变换，传递函数为：

$$H(s)=\frac{K_i}{Js\hat{}\ 2+Cs+K}$$

设J＝0.01 Kg m²,C＝.004 nm/(rad/s),K＝10 nm/rad,及 Ki＝.05 nm/rad。

```
num=Ki;
den=[J C K];
H=tf(num,den);
Pause
clc
```

设计一个数字控制器，能对读写头的位置进行精确的控制。先对连续系统离散化。因为该系统对其输入具有一个数模转换器(零阶保持)，所以可用函数 c2d 的"zoh"离散方法。采样周期 Ts＝0.005s(5ms)。

```
Ts=0.005;
Hd=c2d(H,Ts,'zoh');
```

比较连续和离散的系统 Bode 图(图 9.6-10)：

```
bode(H,'-',Hd,'--')
pause
```

现在分析离散系统，画阶跃响应图(图 9.6-11)。

```
step(Hd),pause
```

由图可看出系统振荡很厉害，这是阻尼太小的缘故，可以通过计算系统的开环特征值来检验这一点，下面求系统的开环离散系统特征值：

```
disp('Open loop discrete eigenvalues'),
```

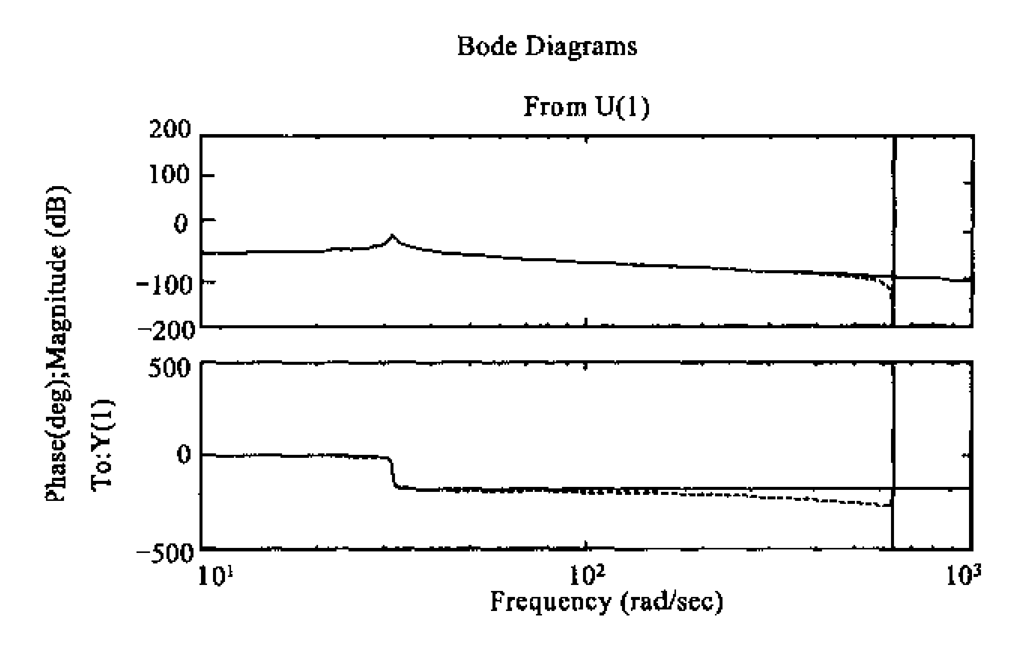


图 9.6-10　连续和离散的 Bode 图

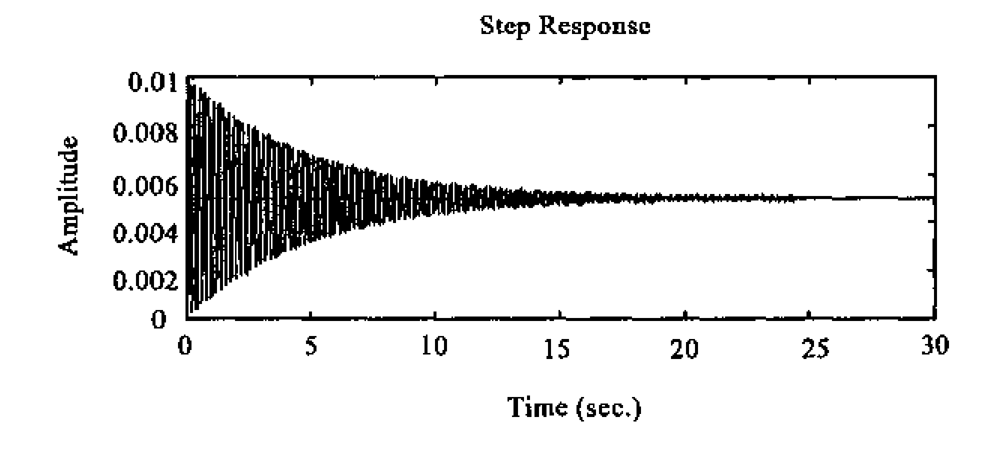


图 9.6-11　系统阶跃响应

```
damp(Hd);
zgrid('new'),pzmap(Hd),hold off
```

图中的"x"号表示开环极点，注意图中的极点离单位圆非常近，说明阻尼很小。因此需要设计一个补偿器以增加系统的阻尼。

```
pause  %画完图按任意键...
clc
```

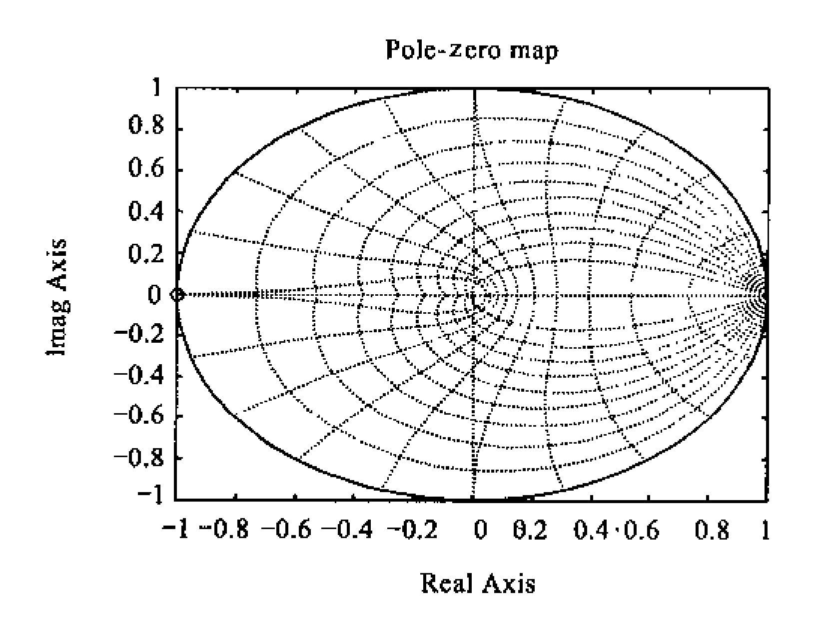


图 9.6-12 系统零极点分布

我们试着设计一个补偿器，最简单的补偿器是增益补偿器。为进行设计，先画出系统的根轨迹图。

```
rlocus(Hd),zgrid,set(gca,'xlim',[-1 1],'ylim',[-1.5,1.5]);
pause
D=zpk(0.85,0,1,Ts)
oloop=Hd D;
bode(Hd,'--',oloop,'-')
```

我们已看到频率响应如何变化(图 9.6-13 a)，带有补偿器的开环响应是一根实线，可见补偿器确实使系统得到了超前校正。

```
Pause
clc
```

现在，我们再来看看带有补偿的根轨迹图。

```
rlocus(oloop);zgrid,set(gca,'xlim',[-1 1],'ylim',[-1.5,1.5]);
pause
```

从图中(图 9.6-13 b)可看出这次极点会停留在单位圆一段时间。现在用 rlocfind 函数选择具有最大阻尼的极点(用 zgrid 函数画出的曲线显示了 z 从 0 到 1 的阻尼，步长为 0.1)。

```
pause % 按任意键后，可在图上选择极点
[k,poles]=rlocfind(oloop);
```

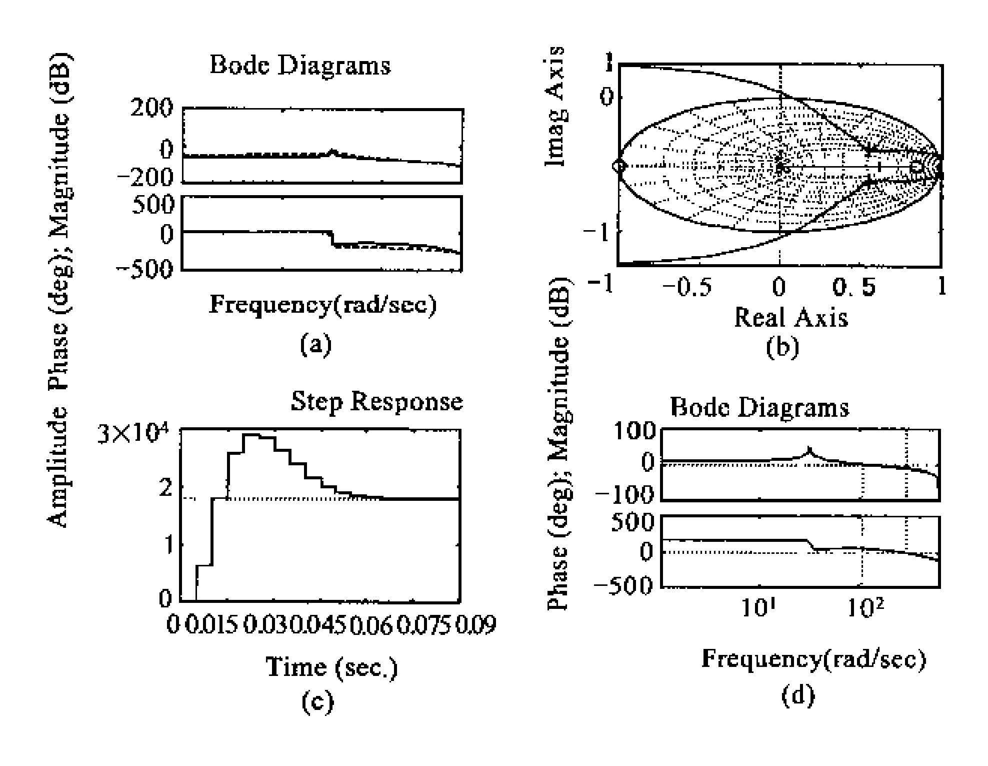


图 9.6-13

此时在上面的根轨迹图上出现一个"+"光标，可用鼠标拖动光标选择任意极点。假设在图中选择了一极点，并按鼠标左键确定，此时图中会出现你所选择的极点坐标：

```
selected=_point=0.0543-0.6262;
```

下面求所选极点形成的闭环系统的增益和闭环极点：

```
disp(['You chose gain: ',num2str(k)]),ddamp(poles,Ts);
```

命令窗上则显示所选增益及相应特征值、特征频率等。然后再利用反馈形成闭环以便分析所进行的设计，此处为正反馈。

```
cloop = feedback(oloop,k);
```

这些系统的特征值与选择相符。下面再求闭环系统的特征值：

```
disp('Closed loop eigenvalues'), damp(cloop)
pause
```

画出闭环系统的阶跃响应。

```
step(cloop), pause
```

可看出所设计系统的响应很理想(图 9.6-13 c)，经过 14 个采样值后过渡过程结束，这个过程经历的时间是 $14\times0.005 = 0.07$ 秒。

```
disp(['Our disc drive will have a seek time > ',num2str(14*Ts),' seconds.'])
pause % Press any key to continue ...
```

```
clc
```

下面我们再看看设计的鲁棒性。最为普通的鲁棒判据是增益裕度和相位裕度。用函数 margin 计算这些裕度。首先将我们的设计系统与单位增益反馈连接起来以形成单位反馈开环系统：

```
olk =k*oloop;
```

下面计算增益裕度 Gm、相角裕度 Pm（单位是度）和相应的转折频率 Wcg 及 Wcp（单位是 radians/sec）：

```
[Gm,Pm,Wcg,Wcp] = margin(olk);
margins = [Gm Wcg Pm Wcp]
```

增益余度（单位是度）：

```
20 * log10(Gm)
```

用这些余度绘制 bode 图(图 9.6-13 d)：

```
margin(olk), pause,echo off
```

可看出设计是鲁棒的，可以有 10 分贝的增益裕度和 40°相角裕度，系统具有一定的稳定裕度。以这种设计方法进行设计，我们会发现能找到一个补偿器使开环系统稳定并减少寻道时间（增大阻尼能减小阶跃响应中的调节时间）。

9.7 比例积分与微分控制

比例、积分与微分控制是控制系统设计中常用的控制律。对于控制系统的设计，可以用这些基本的控制律设计控制器，也可采用由它们组合面成的复合控制律。下面是常用的控制形式：

①比例(P)控制：$K_D=K_I=0$；

②比例积分(PI)控制：$K_D=0$；

③比例微分(PD)控制：$K_I=0$；

④比例积分微分控制(PID)。

比例控制是最简单的控制结构，它可使系统满足某一方面的要求，如增益裕量(GM)、相位裕量(PM)、稳态误差等。在闭环控制中，微分控制能反映误差变化率，超前校正，改善系统的动态性能。若加入积分控制，可减小系统的稳态误差，提高控制精度。PID 控制器可以用于校正、补偿系统，满足控制系统的要求，是过程控制中普遍应用的控制律之一。

9.7.1 ZIEGLER—NICHOLS 方法

PID 控制的标准方程是

$$u(t)=K_p[e(t)+\frac{1}{T_i}\int_0^t e(\tau)\mathrm{d}\tau+T_d\frac{\mathrm{d}e(t)}{\mathrm{d}t}] \tag{9.7.1-1}$$

设计 PID 控制器时，要调整 K_p 等参数，通常不直接采用根轨迹与频率响应的设计方法。Ziegler 与 Nichols 提出了一种简化的 PID 调节器设计方法，此法基于简单的稳定性分析方法。先求出系统根轨迹与虚轴交点处的 K^* 值（临界开环增益）和相应的 ω^*，式(9.7.1-1)中的 K_p, T_i, T_d 可用下式算出：

$$\left.\begin{aligned}K_p &= 0.6K^*\\ T_i &= \frac{\pi}{\omega^*}\\ T_d &= \frac{\pi}{4\omega^*}\end{aligned}\right\} \tag{9.7.1-2}$$

该设计方法在设计过程中没有考虑任何特性要求，但是这种设计方法比较简单，又可以使过程控制器具有较好的工作性能。

【例 9.7.1-1】 被控对象的传递函数为

$$G(s)=\frac{1}{5s^3+8s^2+6s}$$

用 Ziegler—Nichols 方法设计 PID 控制器。

输入如下语句

```
sys=tf([1],[1 5 8 6 0])
rlocus(sys)                %plot the root locus
[k poles]=rlocfind(sys)    %find the K*
w=imag(poles(3))           %find the w *
```

rlocus 函数绘制系统的根轨迹（图 9.7.1-1），rlocfind 函数求出临界开环增益

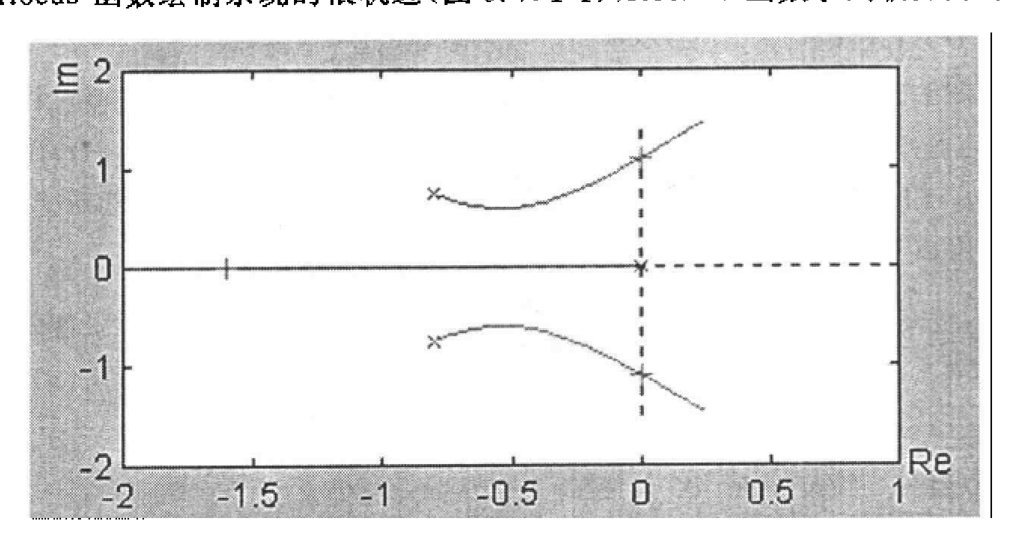


图 9.7.1-1

$K^*=9.5587$ 和相应频率 $\omega^*=1.0938\text{rad/s}$，用式(9.7.1-2)求出 PID 控制器参数：

$K_p=0.6\times9.5587=5.7352$

$T_i=3.14/1.0938=2.8707$

$T_d=3.14/(4\times1.0938)=0.71768$

9.7.2 解 析 法

对于要设计满足一特定闭环系统特性要求的 PID 控制器，可以用解析法，根据给定的稳态误差和操作特性参数来确定 PID 的参数。

图 9.7.2-1 是 PID 控制系统的框图，其开环传递函数为

$$(K_p+K_d s+\frac{K_i}{s})G(s) \tag{9.7.2-1}$$

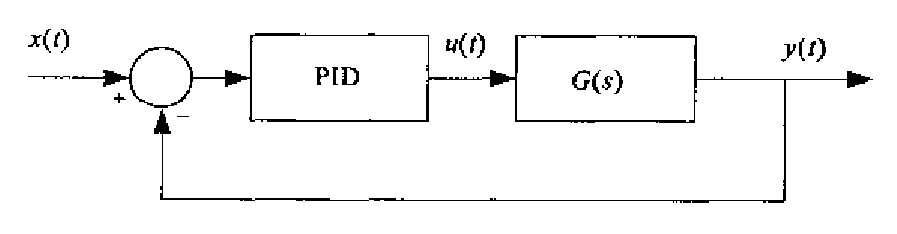


图 9.7.2-1

如果 $G(s)$ 是 n 型系统，补偿后的系统则为 $n+1$ 型系统。误差系数 K_{n+1} 等于稳态误差 e_{ss} 的倒数：

$$K_{n+1}=s^n K_i G(S)\big|_{s=0}=\frac{1}{e_{ss}} \tag{9.7.2-2}$$

如果给出系统的稳态误差，从上式可以求出 K_i。根据超调量和过渡过程时间，可以确定系统闭环阻尼系数 ζ 和自然频率 ω_n。因闭环自然频率 ω_n 对应开环增益穿越频率 ω_c，相角裕量 PM 可以由闭环阻尼系数求出。在 $\omega=\omega_c$ 处，补偿系统的增益为 1，相角 $\theta(\omega_c)=-180°+\text{PM}$。因此由式(9.7.2-1)，有

$$(K_p+j\omega_c K_d+\frac{K_i}{j\omega_c})G(j\omega_c)=1e^{j\theta(\omega_c)} \tag{9.7.2-3}$$

$$K_p+j\omega_c K_d=\frac{1e^{j\theta(\omega_c)}}{G(j\omega_c)}+\frac{jK_i}{\omega_c}=R+jX$$

可以看出　　$K_p=R\quad K_d=X/\omega_c$

【例 9.7.2-1】 设计 PID 控制器的解析方法。仍用【例 9.7.1-1】，系统的技术指标要求如下：

单位斜坡输入稳态误差 0.1；

超调量 15%；

过渡过程时间 2s。

原系统为 I 型系统，加入 PID 控制器后的稳态误差系数

$K_2=sK_iG(s)|_{s=0}=K_i/6=1/0.1$；

$K_i=20/6$。

由超调量和调节过渡过程，可以确定闭环阻尼系数 $\zeta=0.52$，自然频率 $\omega_n=3.38$rad/s。因此 $\omega_c=3.38$rad/s，PM=77°。

有了上面的参数，可以计算 K_p，K_d。输入以下语句：

```
pm=77
wc=3.38
ki=60
num=[1]
den=[5 8 6 0]
numg=polyval(num,j*wc)
deng=polyval(den,j*wc)
g=numg/deng
thetar=(pm-180)*3.14/180
ejthetar=cos(thetar)+j*sin(thetar)
eqn=(ejthetar/g)+j*(ki/wc)
x=imag(eqn)
r=real(eqn)
kp=r
kd=x/wc
```

执行上述语句得　kp=-147.9，kd=43.1。

9.7.3 PD 控制

PD 控制应用非常普遍，下面讨论 PD 控制的解析设计。增加积分环节可以减小系统稳态误差，但增加了极点，使得系统趋于不稳定。增加微分环节，相当于系统增加了零点，可增加系统的稳定性。微分控制器常添加在反馈通道中，特别在希望增加阻尼比的情况。

【例 9.7.3-1】 PD 控制。

被控对象的传递函数如前两例，设计 PD 控制器，以满足下例特性要求：

pm=65°，$\omega_c=2$rad/s

PD 控制器有两个变化参数，用 PID 设计程序，设 $K_i=0$，求得 PD 系数：

$k_p=-11.8944, k_d=20.4115$

由于微分环节的固有噪声问题，一般不构造一个纯微分环节（$K_d s$）。但在许多情况下，微分项总有一个极点与它关联，这时可以把 PD 控制器看作为超前补偿器。PD 控制可以实现直接对输出变化速度的测量（如转速，图 9.7.3-1）。在这种情况下，微分项通常被放在被控对象内环中。

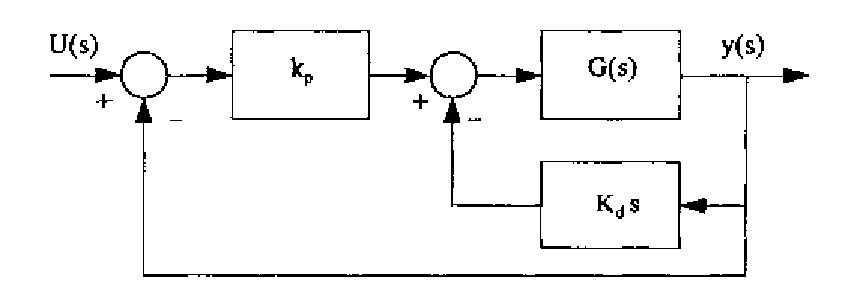


图 9.7.3-1

习 题 九

1. 求下列系统的零、极点，分析稳定性，并说明是否为最小相位系统。

(1) $G(s)=\dfrac{100}{s(s^2+8s+24)}$

(2) $G(s)=\dfrac{3s+1}{s^2(300s^2+600s+50)}$

(3) $G(s)=\dfrac{0.2(s+2)}{s(s+0.5)(s+0.8)(s+3)}$

(4) $\dot{x}=\begin{bmatrix}0 & 1 & -1\\ -6 & -11 & 6\\ -6 & -11 & 5\end{bmatrix}x+\begin{bmatrix}0\\0\\1\end{bmatrix}u \qquad y=[1 \quad 0 \quad 0]x$

2. 求可控、可观测的约当标准型。

(1) $\dot{x}=\begin{bmatrix}-2 & 2 & -1\\ 0 & -2 & 0\\ 1 & -4 & 0\end{bmatrix}x+\begin{bmatrix}0\\0\\1\end{bmatrix}u \qquad y=[1 \quad -1 \quad 1]$

(2) $\dot{x}=\begin{bmatrix}1 & 0 & 0 & 0\\ 2 & -3 & 0 & 0\\ 1 & 0 & -2 & 0\\ 4 & -1 & -2 & -4\end{bmatrix}x+\begin{bmatrix}0\\0\\1\\2\end{bmatrix}u \qquad y=[3 \quad 0 \quad 1 \quad 0]x$

3. 分别用 $T=0.01$，$T=1$ 的采样周期，将 2 题中(1)(2)题离散化，得出离散方程后，再将其转化为连续模型，并比较两种采样周期下变换的结果。

4. 绘制下面各系统的响应曲线，包括 Bode 图和 Nyquist 图，并求其幅值裕量和相位裕量。

(1) $G(s)=\dfrac{100}{s^2+10s+1000}$

(2) $G(s)=\dfrac{8}{s^2+2s+8}$

(3) 单位反馈系统的开环传递函数

a. $G(s)=\dfrac{10(s+1)}{s(s-1)(s+5)}$

b. $G(s)=\dfrac{10}{s(s+1)(2s+3)}$

5. 绘制 4 题中(1)(2)(3)题的阶跃响应和脉冲响应曲线。

第十章　信号处理

10.1　信号处理的基本概念

10.1.1　离散时间信号的MATLAB表示

在数字信号处理(DSP)中,离散时间信号用序列 $x(n)$ 表示,其中 n 是采样数(可以定义 n 的取值范围),因此信号处理的数据对象(一维或二维信号序列,多路信号)可以方便地用MATLAB的矩阵数据格式表示。

在MATLAB中表示一个有六个元素的一维信号序列,可用命令

x=[7 4 8 −3 2 5].

x转置后成为列向量。用列向量表示一维信号更好一些,因为这样有利于用矩阵表示多路信号。用矩阵表示多路信号时,列表示一路信号,行表示各路信号的采样值。如三路信号系统 x,2x,x/factor(factor是比例因子),可以输入

```
    y = [x 2 * x x/factor]
y =
      7.0000     14.0000     5.8333
      4.0000      8.0000     3.3333
      8.0000     16.0000     6.6667
     -3.0000     -6.0000    -2.5000
      2.0000      4.0000     1.6667
      5.0000     10.0000     4.1667
```

10.1.2　典型离散信号

1. 单位采样信号

$$\delta(n)=\begin{cases}1 & n=0\\ 0 & n\neq 0\end{cases} \qquad (10.1.2\text{-}1)$$

若构造有 $N=100$ 个数据的序列,单位采样信号可用语句

```
y = [1; zeros(N-1,1)];    % impulse
```

表示,也可用数组操作语句

```
n = [1:N]'
```

$$\delta(n-k)=\begin{cases}0 & a\leqslant n<k\\ 1 & n=k\\ 0 & k<n\leqslant b\end{cases} \qquad (10.1.2\text{-}2)$$

可以输入

```
k = 5
y =((n-k)==0)   %here a=1 b=N
```

2. 单位阶跃序列

$$u(n)=\begin{cases}1 & n\geqslant 0\\ 0 & n<0\end{cases} \qquad (10.1.2\text{-}3)$$

用MATLAB语句表示

```
y = ones(N,1); % step (filter assumes 0 initial cond.)
```

如果

$$u(n-k)=\begin{cases}1 & n\geqslant k\\ 0 & n<k\end{cases} \qquad (10.1.2\text{-}4)$$

可以输入

```
k = 5
y =((n-k)>= 0)
```

3. 正弦序列

$$x(n)=A\sin(\omega n+\varphi) \qquad (10.1.2\text{-}5)$$

设A=1,在MATLAB中可表示为

```
y = sin(w * n+phi)
```

4. 复正弦序列

$$x(n)=e^{j\omega n}=\mathrm{con}(\omega n)+\mathrm{i}\sin(\omega n) \qquad (10.1.2\text{-}6)$$

用MATLAB语句表示

```
y = exp((w * i) * n)
```

5. 指数序列

$$x(n)=a^{|n|} \qquad a\text{ 为常数,且绝对值小于 }1 \qquad (10.1.2\text{-}7)$$

用MATLAB语句表示

```
y = a.^(abs(n))
```

6. 组合信号

MATLAB用于信号处理的函数,在使用时,多数需要一个表示时基的向量,如果以1KHz频率采样,可设置时间向量

```
t = (0:0.001:1)';
```

下面语句产生由t形成的、包含50Hz和120 Hz正弦信号的序列:

```
y = sin(2 * pi * 50 * t)+2 * sin(2 * pi * 120 * t);
```

还可在y中加入随机信号：

```
yn = y + 0.2 * randn(size(t));
plot(t(1:50),yn(1:50))
```

图10.1.2-1是yn前50个数据的图形。

```
y1 = t; % ramp
y2= t.^2;
y3 = square(4 * 2 * pi * t);   %4 Hz 的方波
```

由y1,y2,y3合成的多通道信号

```
z = [y1 y2 y3];
```

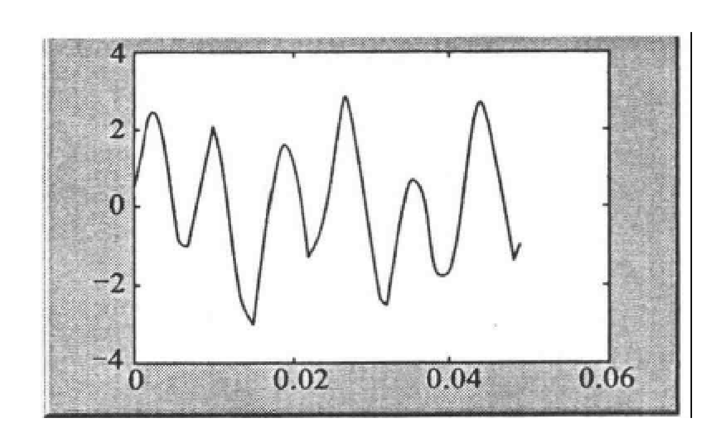


图10.1.2-1　yn前50个数据的图形

10.1.3　离散信号的基本运算

1. 信号延迟

给定离散信号 $x(n)$，若信号

$$y(n) = x(n-k) \tag{10.1.3-1}$$

$y(n)$是$x(n)$在数轴右移k个采样周期得到的序列。对序列$x(n)$在k时刻的值$x(k)$可用$\delta(n)$的延迟表示：

$$x(k) = x(n)\delta(n-k)$$

这是$\delta(n)$的"采样"性质。$x(n)$在n的所有时刻的值可表示为

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k) \tag{10.1.3-2}$$

2. 信号相加和相乘

设有两个信号序列$x_1(n)$，$x_2(n)$，信号相加和相乘表示在相同时刻t的值相加或相乘。

(1)相加

$y(n)=x_1(n)+x_2(n)$

MATLAB 中用

y=x1+x2

表示，注意满足数组运算条件。

(2)相乘

$y(n)=x_1(n)x_2(n)$

用点乘语句

y=x1. * x2

表示，注意满足数组运算条件。

(3)标量乘

$y(n)=ax(n)$

用语句 y=a * x 表示。

(4)采样和

$$y=\sum_{n=a}^{b}x(n)$$

用函数 sum

y = sum(x(a:b))表示。

(5)采样积

$$y=\prod_{n=a}^{b}x(n)$$

用语句 y = prod(x(a:b))求采样积。

10.1.4　离散 LSI 系统的输入输出关系

一个离散时间系统，当输入 $x(n)=\delta(n)$时，称输出 $y(n)$是系统的单位采样响应，记为 $h(n)$，此时 $y(n)=h(n)$。由式(10.1.3-2)，输入信号 $x(n)$可表示为 $\delta(n)$及其移位的线性组合

$$x(n)=\sum_{k=-\infty}^{\infty}x(k)\delta(n-k)$$

根据 LSI 的移不变性，系统对 $x(n)$的响应

$$y(n)=\sum_{k=-\infty}^{\infty}x(k)h(n-k) \tag{10.1.4-1}$$

上式是 LSI 系统的卷积，简记为 $y(n)=x(n)\cdot h(n)$，由 LSI 的因果性，上式可写作：

$$y(n) = \sum_{k=0}^{\infty} x(k)h(n-k)$$

MATLAB 有求卷积的函数 conv。

```
x = randn(5,1); % a random vector of length 5
h = [1 1 1 1]/4; % length 4 averaging filter
y = conv(x,h);
```

可求系统的卷积。

10.1.5 离散傅里叶变换(DFT)

离散傅里叶变换对

$$\left.\begin{aligned} X(k) &= \sum_{n=0}^{N-1} x(n)e^{-j\frac{2\pi}{N}nk} = \sum_{n=0}^{N-1} x(n)W_N^{nk} \quad k = 0,1,\cdots,N-1 \\ x(n) &= \frac{1}{N}\sum_{n=0}^{N-1} X(k)e^{j\frac{2\pi}{N}nk} = \frac{1}{N}\sum_{n=0}^{N-1} X(k)W_N^{-nk} \quad k = 0,1,\cdots,N-1 \end{aligned}\right\} \quad (10.1.5\text{-}1)$$

式中 $W_N = e^{-j\frac{2\pi}{N}}$。实际处理时，若 $x(n)$ 是有限长序列，可以令其长度为 N，如果 $x(n)$ 是无限长序列，可以加窗截成 N 点，再用式(10.1.5-1)求 $x(n)$ 的频谱。在 MATLAB 中，用函数 fft，ifft 可以很容易地实现 DFT 和 IDFT。

10.1.6 DFT 特性

1. 线性

若 $x_1(n)$，$x_2(n)$ 都是 N 点序列，其 DFT 分别是 $X_1(k)$，$X_2(k)$，则

$$\text{DFT}[ax_1(n) + bx_2(n)] = aX_1(k) + bX_2(k) \quad (10.1.6\text{-}1)$$

2. 移位性质

将 N 点序列 $x(n)$ 左移或右移 m 个采样周期，有

$$\text{DFT}[x(n+m)] = W^{-km}X(k) \quad (10.1.6\text{-}2a)$$

$$\text{DFT}[x(n-m)] = W^{km}X(k) \quad (10.1.6\text{-}2b)$$

3. 奇偶虚实对称性质

(1)若 $x(n)$ 为复序列，其 DFT 为 $X(k)$，则

$$\text{DFT}[x*(n)] = X*(-k) \quad (10.1.6\text{-}3)$$

(2)若 $x(n)$ 为实序列，则

$$X*(k) = X(-k) = X(N-k) \quad (10.1.6\text{-}4a)$$

$$X_R(k) = X_R(-k) = X_R(N-k) \quad (10.1.6\text{-}4b)$$

$$X_I(k) = -X_I(-k) = -X_I(N-k) \quad (10.1.6\text{-}4c)$$

$$|X(k)| = |X(N-k)| \quad (10.1.6\text{-}4d)$$

$$\arg[X(k)] = -\arg[X(-k)] \tag{10.1.6-4e}$$

(3)若 $x(n)$是偶序列，即 $x(n)=x(-n)$，则 $X(k)$是实序列。

(4)若 $x(n)$是奇序列，即 $x(n)=-x(-n)$，则 $X(k)$是纯虚序列。

4. Parseval(巴斯瓦)定理

$$\sum_{n=0}^{N-1}|x(n)|^2 = \frac{1}{N}\sum_{k=0}^{N-1}|X(k)|^2 \tag{10.1.6-5}$$

巴斯瓦定理反映了信号在一个域及其对应的变换域中的能量守恒原理。

5. 循环卷积

设 $x(n)$，$h(n)$，$y(n)$都是 N 点序列，其 DFT 分别是 $X(k)$，$H(k)$，$Y(k)$。$x(n)$和 $h(n)$的循环卷积：

(1)时域循环卷积

$$\left.\begin{aligned} y(n) &= x(n)\otimes h(n) \\ Y(k) &= X(k)H(k) \end{aligned}\right\} \tag{10.1.6-6}$$

式中 $x(n)\otimes h(n) = \sum_{i=0}^{N-1}x(i)h(n-i)$，$\otimes$表示作 N 点循环。

(2)频域循环卷积

$$\left.\begin{aligned} y(n) &= x(n)h(n) \\ Y(k) &= X(k)\otimes H(k) \end{aligned}\right\} \tag{10.1.6-7}$$

10.1.7 利用DFT计算线性卷积

从前面的讨论，我们知道DFT 得到的是循环卷积，下面讨论如何用DFT 计算两个序列的线性卷积。

设 $x(n)$是 M 点序列，$h(n)$是 L 点序列，那么它们的线性卷积 $y(n)=x(n)h(n)$是 $M+L-1$ 点的序列。因此 $y(n)$的 DFT—$Y(k)$必须是 $M+L-1$ 点序列，相应的 $X(k)$，$H(k)$也是 $M+L-1$ 点序列，$X(k)$，$H(k)$对应的时域序列 $x(n)$，$h(n)$也必须是 $M+L-1$ 点序列。只有这样，由 $Y(k)$作逆变换 $y(n)=\mathrm{IDFT}[Y(k)]$所得 $y(n)$才能保证是 $x(n)$，$h(n)$的线性卷积。计算步骤：

1. 扩展 M 点序列 $x(n)$，L 点序列 $h(n)$，构造新序列 $x'(n)$，$h'(n)$，它们的长度都是 $M+L-1$。

$$x'(n) = \begin{cases} x(n) & n = 0,1,2,\cdots,M-1 \\ 0 & n = M,\cdots,M+L-2 \end{cases} \tag{10.1.7-1}$$

$$h'(n) = \begin{cases} h(n) & n = 0,1,2,\cdots,L-1 \\ 0 & n = L,\cdots,M+L-2 \end{cases} \tag{10.1.7-2}$$

2. 用(10.1.6-6)式计算

$y'(n)=x'(n)\otimes h'(n)$，$N=M+L-1$

而 $y(n)=y'(n)=x(n)h(n)$。

3. 用 DFT 求 $y(n)$，则

$y(n)=y'(n)=\mathrm{IDFT}[X'(k)\ H'(n)]$

$X'(k)$，$H'(n)$是 $x'(n)$，$h'(n)$的 DFT。

10.2 数字滤波器分析与实现

数字滤波器输出 $y(n)$的 z 变换：

$$Y(z)=H(z)X(z)=\frac{b(1)+b(2)z^{-1}+\cdots+b(M+1)z^{-M}}{a(1)+a(2)z^{-1}+\cdots+a(N+1)z^{-N}}X(z) \tag{10.2-1}$$

式中 $H(z)$是滤波器传递函数。滤波器阶数取 M，N 中大者。若 $M=0$，称滤波器为无限脉冲响应滤波器(IIR)或自回归滤波器(AR)。若 $N=0$，则称为有限脉冲响应滤波器(FIR)或滑动平均滤波器(MR)。若 M，N 都大于零，亦为无限脉冲响应滤波器，在随机过程中称自回归滑动平均滤波器(ARMA)。

将(10.2-1)式两边同乘右边分母，进行反 z 变换，归一化系数 $a(1)$，得数字滤波器差分方程

$$y(n)=\sum_{j=0}^{M}b(j+1)x(n-j)-\sum_{i=1}^{N}a(i+1)y(n-i) \tag{10.2-2}$$

10.2.1 IIR 滤波器

1. IIR 滤波器的直接实现

展开式(10.2-2)：

$y(n)=b(1)x(n)+b(2)x(n-1)+\cdots+b(M+1)x(n-M)-a(2)y(n-1)-\cdots-a(N+1)y(n-N)$

图 10.2.1-1 的数字网络是 IIR 滤波器的直接实现。由于加法器的作用，每个系数的量化误差和乘法器的舍入误差会对输出产生累积效应，系统阶次越高，结果误差越大，这是直接实现的缺点。实际应用中，对于高阶系统，应避免采用这种方式实现 IIR。一般采用由一阶、二阶系统构成的串联或并联方式更好一些。

2. IIR 滤波器的级联实现

设 $N\geqslant M$，N 是偶数，线性系统的传递函数 $H(z)$可表示为二阶系统的连乘

$$H(z)=\prod_{k=1}^{L}H_k(z)=\prod_{k=1}^{L}\frac{b_k(1)+b_k(2)z^{-1}+b_k(3)z^{-2}}{a_k(1)+a_k(2)z^{-1}+a_k(3)z^{-2}} \tag{10.2.1-1}$$

式中 $L=N/2$。若 N 是奇数，则 $L=(N+1)/2$，有一个一阶系统。图 10.2.1-

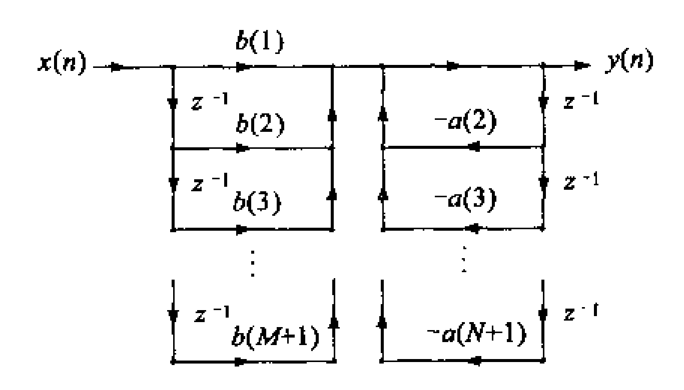


图 10.2.1-1　IIR 滤波器的数字网络

2 是 $a_k(1)$ 归一化后的二阶系统 $H_k(z)$ 的数字网络。IIR 的输出 $y(n)$ 是 L 个这样的网络的级联。

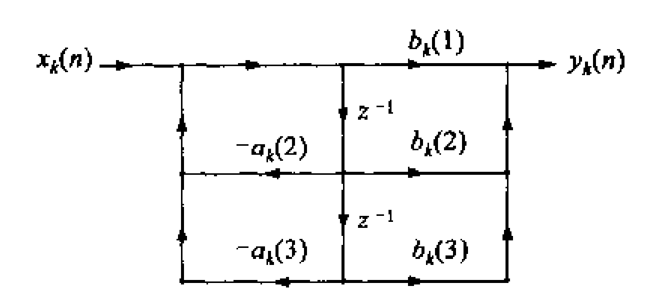


图 10.2.1-2　二阶系统数字网络

3. IIR 滤波器的并联实现

$H(z)$ 还可分解成如下形式(归一化 $a(1)$)

$$H(z)=\sum_{i=1}^{L1}\frac{A_i}{1+p_i z^{-1}}+\sum_{j=1}^{L2}\frac{b_j(1)+b_j(2)z^{-1}}{1+a_j(1)z^{-1}+a_j(2)z^{-2}} \tag{10.2.1-2}$$

$H(z)$ 化为 $L1+L2$ 个 $H_i(z)$，系统有一个输入 $x(n)$，$L1+L2$ 个输出 $y_i(n)$，系统总输出 $y(n)$ 是它们的和

$$y(n)=\sum_{i=1}^{L1+L2}[h_i(n)x(n)] \tag{10.2.1-3}$$

各子系统间是并联关系，图 10.2.1-3 是并联 IIR 系统的数字网络。并联结构的每个子系统的系统量化误差和舍入误差仅影响本系统，与其他系统无关，累积误差较前两种形式小。

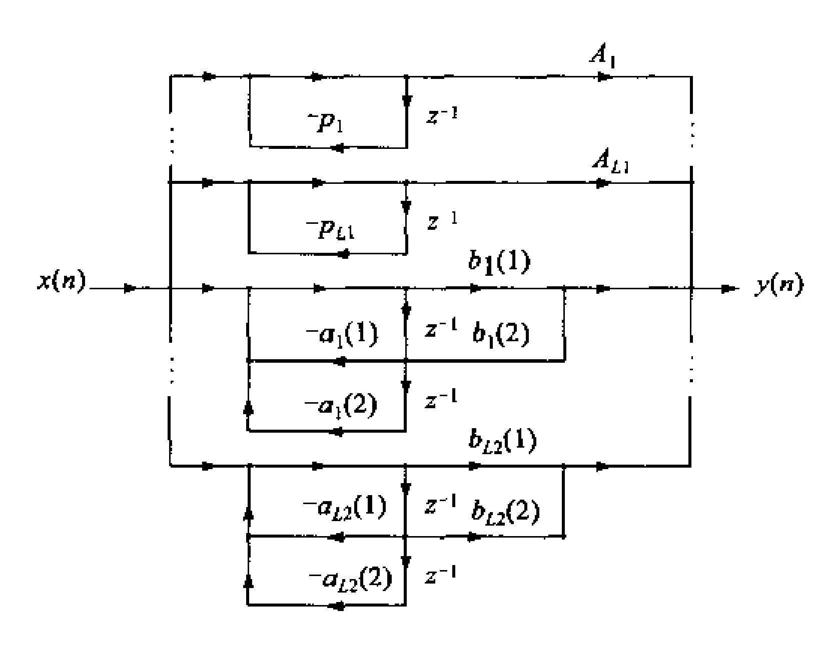


图 10.2.1-3　并联 IIR 系统的数字网络

10.2.2　FIR 滤波器结构

当 $N=0$ 时，10.2-1 式的 $H(z)$ 有如下形式

$$H(z)=b(1)+b(2)z^{-1}+\cdots+b(M+1)z^{-M} \qquad (10.2.2\text{-}1)$$

此即 FIR 滤波器的差分方程。同 IIR 一样，FIR 可以有直接形式和级联形式，并联形式较少用于 FIR。

图 10.2.2-1　直接 FIR 系统数字网络

1. FIR 的直接形式和级联形式

图 10.2.2-1 是式(10.2.2-1)的数字网络。

FIR 系统 $H(z)$ 的二阶分解

$$H(z)=\prod_{i=1}^{L}[b_i(1)+b_i(2)z^{-1}+b_i(3)z^{-2}] \qquad (10.2.2\text{-}2)$$

$L=M/2$，若 M 是奇数，$L=(M+1)/2$，其中有一个一阶系统。图 10.2.2-2 是级联 FIR 系统数字网络。

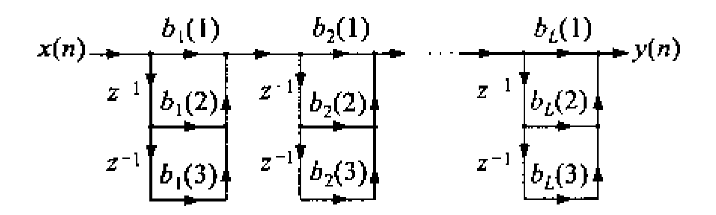


图 10.2.2-2 级联 FIR 系统数字网络

2. FIR 的线性相位结构

当 $h(n)=\pm h(M-n)$，FIR 滤波器具有线性相位。图 10.2.2 3，图 10.2.2-4 分别是 M 为奇数和偶数的 FIR 系统数字网络。实现具有线性相位结构的 FIR 时，比较容易(图 10.2.2-1)，少进行 $M/2$ 次乘法运算。

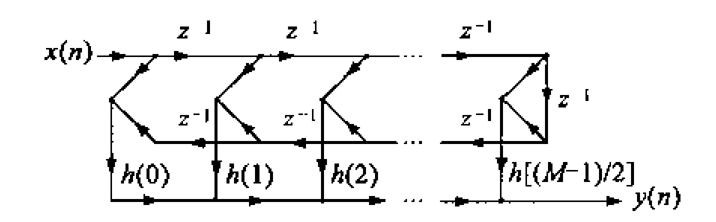


图 10.2.2-3 线性 FIR 系统数字网络(M 为奇数)

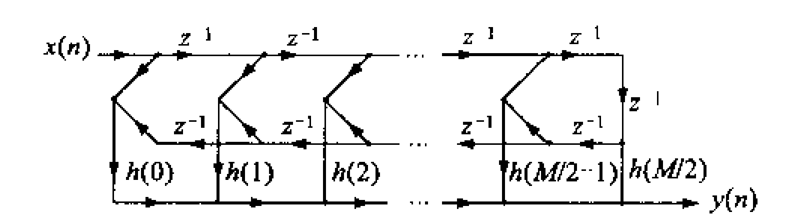


图 10.2.2-4 线性 FIR 系统数字网络(M 为偶数)

3. FIR 系统的频率采样实现

若 FIR 系统脉冲响应 $h(n)$ 的 N 点 DFT 为 $H(k)(0\leqslant k\leqslant N-1)$，可以用 $H(k)$ 表示 $h(n)$ 的 z 变换 $H(z)$：

$$H(z)=Z[h(n)]=Z[\mathrm{IDFT}(H(k))] \tag{10.2.2-3}$$

$$H(z)=\left(\frac{1-z^{-N}}{N}\right)\sum_{k=0}^{N-1}\frac{H(k)}{1-W_N^{-k}z^{-1}} \tag{10.2.2-4}$$

式(10.2.2-4)表示 FIR 的频率采样结构，这种结构分成两部分：第一部分是

$(1-z^{-N})/N$，第二部分由 N 个 $H(k)/(1-W_N^{-k}z^{-1})$ 并联而成。图 10.2.2-5 是FIR系统的频率采样实现。

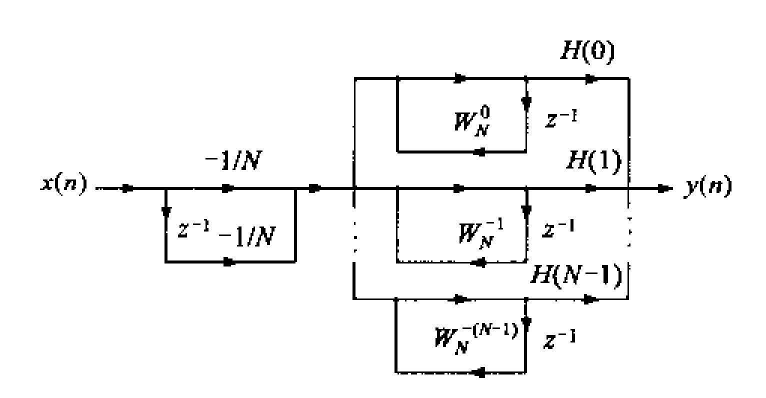


图 10.2.2-5 FIR 系统的频率采样数字网络

10.2.3 格形滤波器结构

格形(Lattice)滤波器在谱分析、语音处理及自适应滤波等领域中已有广泛的应用，下面讨论全零点格形滤波器和全极点格形滤波器。

1. 全零点(FIR)格形滤波器

设(10.2.2-1)式的 $b(1)=1$，M 阶的 FIR 滤波器的传递函数

$$H(z) = B(z) = 1 + \sum_{i=1}^{M} b(i)z^{-i} \qquad (10.2.3\text{-}1)$$

其 M 阶格形结构数字网络如图 10.2.3-1 所示。

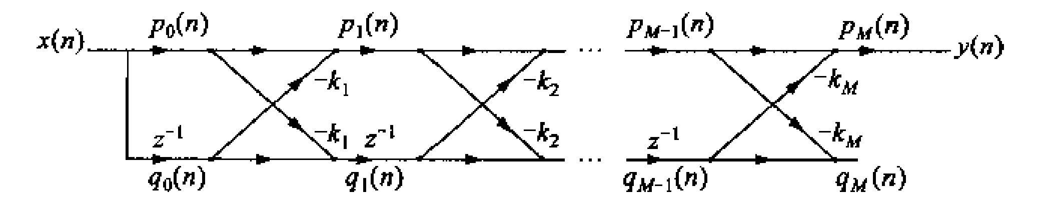


图 10.2.3-1 FIR 格形数字网络

滤波器数字网络的每一阶有以下递推关系

$$p_m(n) = p_{m-1}(n) - k_m q_{m-1}(n-1) \qquad (10.2.3\text{-}2a)$$

$$q_m(n) = -k_m p_{m-1}(n) + q_{m-1}(n-1) \quad m = 1,2,\cdots,M \qquad (10.2.3\text{-}2b)$$

$$p_0(n) = q_0(n) = x(n) \qquad (10.2.3\text{-}2c)$$

$$y(n) = p_M(n) \qquad (10.2.3\text{-}2d)$$

如果定义

$$B(z)=\frac{P_m(z)}{P_0(z)}=1+\sum_{i=1}^{m}b_m(i)z^{-i}\qquad m=1,2,\cdots,M \quad (10.2.3\text{-}3)$$

系数 $b_m(i)$ 表示 m 阶 FIR 系统的第 i 个系数。下面给出 k_m 和滤波器系数的递推关系。

$$b_m(m)=-k_m \quad (10.2.3\text{-}4a)$$

$$b_m(i)=b_{m-1}(i)-k_m b_{m-1}(m-i) \quad (10.2.3\text{-}4b)$$

$$k_m=-b_m(m) \quad (10.2.3\text{-}5a)$$

$$b_{m-1}(i)=[b_m(i)+k_m b_m(m-i)]/(1-k_m^2) \quad (10.2.3\text{-}5b)$$

式中 $i=1,2,\cdots,m-1$ ，$m=1,2,\cdots,M$。实际运用时，一般给出 $H(z)=B(z)=B_m(z)$，这样即可以用式(10.2.3-2)，(10.2.3-4)，(10.2.3-5)求出全零点(FIR)格形滤波器的有关系数。注意，若 $|k_m|=1$，不能用格形结构实现 FIR 滤波器。

2. 全极点(IIR)格形滤波器

IIR 滤波器的全极点传递函数

$$H(z)=\frac{1}{1+\sum_{i=1}^{M}a_M(i)z^{-i}} \quad (10.2.3\text{-}6)$$

其格形结构(图 10.2.3-2)是图 10.2.3-1 的逆，系数 k_i，$a_m(i)$ 求法与 FIR 一样，只是把 $b_m(i)$ 换成了 $a_m(i)$。

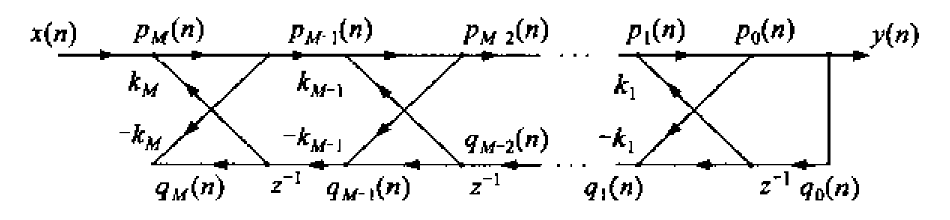


图 10.2.3-2 IIR 格形数字网络

10.2.4 滤波函数

1. filter

对于已知式(10.2-1)的 $H(z)$ 的数字滤波器，用 filter 函数可以产生它的输出，filter 调用格式：

```
y = filter(b,a,x)
```

参数 b 是传递函数分子多项式系数，a 是传递函数分母多项式系数，x 为输入向量。

```
a = [1.0000 -0.3695 0.1958];
b = [0.2066 0.4131 0.2066];
x = [1 zeros(1,100)];
y = filter(b,a,x);
```

对于分组的数据，其初值和终值是很重要的，特别是要考虑计算机存储限制时。设有两组5000点的数据：

```
x1 = randn(5000,1); % two random sequences to
x2 = randn(5000,1); % serve as simulated data
```

x1是最初10秒钟采集的，x2是后10秒钟的。数据长度

```
x = [x1; x2]
```

如果没有足够的存储空间能够顺序地一次对x滤波，可以用x1的终值作为x2滤波的初值，以保证滤波的连续性。

```
[y1,zf] = filter(b,a,x1);
y2 = filter(b,a,x2,zf);
```

zf是第一条语句输出的终值，在后一条语句中，作为初值参数。

2. filtic

函数filtic为filter返回初值。filtic计算延迟向量，使滤波器能反映过去的输入和输出：

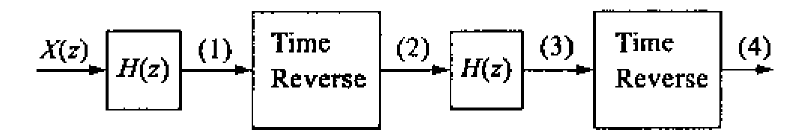
```
zf = filtic(b,a,flipud(y1),flipud(x1));
```

filtic返回与前例相同的延迟向量zf。对较短数据滤波时，filtic可以给出合适的初始条件，这有助于减小瞬态效应。

3. filtfilt

对于FIR滤波器，进行数据滤波时(用filter,conv函数)，只需用固定的采样点延迟输出，就可能设计出线性相位滤波器，而IIR的相位失真通常是强非线性的，为了消除相位失真，可以用filtfilt。filtfilt滤波时使用当前信号的前后数据。由于用了将来数据，因此是非因果的。

现在说明filtfilt如何消除相位失真。filtfilt滤波过程见图10.2.4-1。



(1) $X(z)H(z)$ (2) $X(1/z)H(1/z)$ (3) $X(1/z)H(1/z)H(z)$ (4) $X(z)H(1/z)H(z)$

图10.2.4-1

设序列 $x(n)$ 的 z 变换是 $X(z)$，$x(-n)$ 的 z 变换是 $X(1/z)$，当 $|z|=1$，即 $z=e^{j\omega}$ 时，输出是 $X(e^{j\omega})|H(e^{j\omega})|^2$。这时对于给定的序列 $x(n)$ 的所有采样值，可以得到零相位失真的双线性滤波。

【例 10.2.4-1】 有一由两个分别为 3Hz，40 Hz 的正弦信号合成的序列，采样频率 100Hz。设计一个 9 阶滤波器，比较用 filter，filtfilt 滤波的结果。

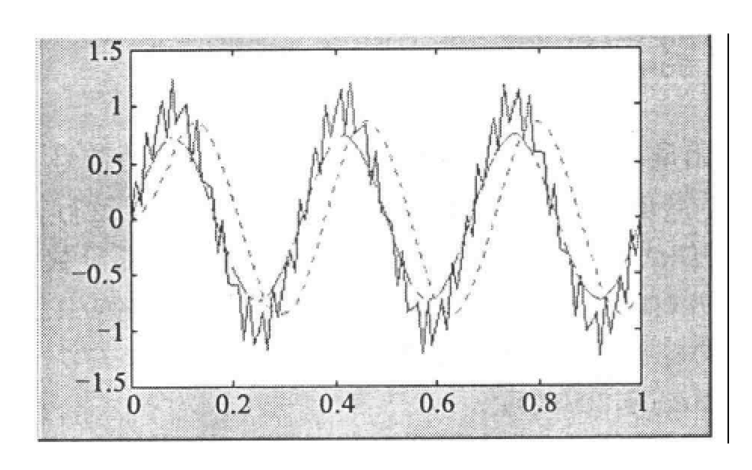


图 10.2.4-2

```
Fs = 100;
t = 0:1/Fs:1;
x = sin(2 * pi * t * 3)+0.25 * sin(2 * pi * t * 40);
b = ones(1,10)/10; % 10 point averaging filter
y = filtfilt(b,1,x); % non-causal filtering
yy = filter(b,1,x); % normal filtering
plot(t,x,t,y,'--',t,yy,':')
```

图 10.2.4-2 是原信号（实线）和滤波后的图形。可以看出滤波后，原信号中 40Hz 频率的分量被消除了。虚线(filtfilt)与原信号的相位一致，点线(filter)大约延迟 5 个采样单位，由于 filtfilt 的幅值平方效应，其增益较 filter 小。

4. fft，fftfilt

我们知道信号处理中系统的输出是输入信号和滤波器脉冲响应卷积的代数和式(10.2-2)，因此可以用 fft 实现滤波。

```
a=1;
n = length(x);
y = ifft(fft(x). * fft(b,n). /fft(a,n));
y=real(y);
```

执行上列语句的结果与前面用 filter 是一样的，只是初始响应不一样。对于长

序列，这种算法效率很低。但对于FIR滤波，可以把长序列截短到适合采用FFT的长度。

y = fftfilt(b,x)

fftfilt 分序列成适当长度，用叠加算法对长序列滤波，这条语句的输出与 filter(b,1,x)一样。

10.3 滤波器设计

滤波器的作用是从已有的信号中去除某一频率范围的信号，只允许指定频率范围的信号通过。简单的滤波，仅滤除指定频率范围的信号就行。有些应用对滤波器有更高的要求，如滤波后信号的衰减、过渡带宽度、滤波器阶数等。

对于简单滤波(如要求从采样频率是100Hz的信号序列中去掉30Hz以上的信号)，用巴特沃思滤波器就能满足要求。

```
[b,a] = butter(5,30/50);
y = filter(b,a,x);
```

上面第一条语句构造一个五阶、截止频率30Hz的巴特沃思滤波器。butter的第二个参数是滤波器的截止频率。x仍用前面数据。

注意：在MATLAB中，滤波器设计函数并不直接用滤波器的截止频率作为参数，而是用规一化的频率作为函数的截止频率参数(前面butter的截止频率就是规一化的)。按惯例，用奈氏频率(采样频率的一半)作为频率单位，因此，规一化频率$0 \leqslant f \leqslant 1$。对于采样频率100Hz的系统，30Hz的截止频率规一化后，等于30/50=0.6奈氏频率。规一化频率乘以π，即被转换成单位圆上的角频率，规一化频率乘以采样频率的一半，就是以Hz为单位的截止频率。

对于设计各种指定性能指标的滤波器，如滤波后的增益、过渡带宽度、线性相位，可以根据实际情况，设计IIR滤波器或FIR滤波器，满足滤波要求。

10.3.1 IIR 滤波器

相对FIR滤波器，IIR可用较低的阶数获得更好的选择性，因此在数据存储和算法上比较经济。但IIR滤波器存在相位非线性，选择性越好，相位非线性越严重，这是其缺陷。对于IIR滤波器的相位非线性，MATLAB可以通过零相位滤波(filtfilt函数)，预处理数据序列，减小IIR滤波器的非线性相位失真。

经典IIR滤波器，如巴特沃思、契比雪夫Ⅰ型、契比雪夫Ⅱ型、贝塞尔以及椭圆滤波器，都以不同逼近准则去近似理想滤波器。MATLAB信号处理工具箱提供了设计这些滤波器(数字的或模拟的)的函数，对于指定技术指标的IIR滤波器，稍作

处理，也能设计出来。

用 yulewalk 设计滤波器无需在连续空间和离散空间转换，其构造的滤波器的幅值响应可以逼近任意所期望的函数，是设计多带带通滤波器的方法之一。

也可用参数建模或系统辨识函数设计 IIR 滤波器。在"系统模型参数估计"部分将说明其设计方法。

表 10-1 描述了设计 IIR 滤波器的方法和 MATLAB 信号处理工具箱中用于设计 IIR 滤波器的函数。

表 10-1

方　法	说　明	MATLAB 函数
模拟原形	将欲设计的数字滤波器的技术指标转换成模拟低通滤波器的技术指标。在连续频域设计模拟低通滤波器，通过频率转换和离散化，得到数字滤波器。	设计低通滤波器： besself，butter，cheby1，cheby2，ellip 估计阶次： buttord，cheb1ord，cheb2ord，llipord 设计低通模拟原形滤波器： besselap，buttap，cheb1ap，cheb2ap，ellipap 频率转换： lp2bp，lp2bs，lp2hp，lp2lp 滤波器离散化： bilinear，impinvar
直接设计	在离散域中直接设计数字滤波器。	yulewalk
参数建模	设计近似描述时域和频域响应的数字滤波器。	频域模型： lpc，prony，stmcb 时域模型： invfreqs，invfreqz

1. 通过模拟滤波器原型设计 IIR 滤波器

基于模拟低通滤波器的原型设计数字 IIR 滤波器，是设计数字 IIR 滤波器的主要方法，下面介绍如何通过原型滤波器，设计出所要求的数字 IIR 滤波器。

(1)设计经典 IIR 滤波器

使用表 10-2 中的滤波器设计函数，很容易设计任意阶的低通、高通、带通和带阻滤波器。表中各函数参数在缺省条件下，返回低通滤波器。注意截止频率参数 Wn 是规一化频率。options 是字符串选项'type'和's'。对于高通滤波器，type=

high,带阻滤波器则type=stop。当设计带通或带阻滤波器时.Wn是二元素向量,分别表示通频带上、下截止频率。若函数增加参数's',则返回模拟IIR滤波器。注意,besself没有数字形式,只能构造模拟IIR滤波器。

表10-2

滤波器类型	设计函数
巴特沃思	[b,a] = butter(n,Wn,options) [z,p,k] = butter(n,Wn,options) [A,B,C,D] = butter(n,Wn,options)
契比雪夫I型	[b,a] = cheby1(n,Rp,Wn,options) [z,p,k] = cheby1(n,Rp,Wn,options) [A,B,C,D] = cheby1(n,Rp,Wn,options)
契比雪夫II型	[b,a] = cheby2(n,Rs,Wn,options) [z,p,k] = cheby2(n,Rs,Wn,options) [A,B,C,D] = cheby2(n,Rs,Wn,options)
椭圆	[b,a] = ellip(n,Rp,Rs,Wn,options) [z,p,k] = ellip(n,Rp,Rs,Wn,options) [A,B,C,D] = ellip(n,Rp,Rs,Wn,options)
贝塞尔(模拟)	[b,a] = besself(n,Wn,options) [z,p,k] = besself(n,Wn,options) [A,B,C,D] = besself(n,Wn,options)

下面是设计例子:

```
[b,a] = butter(9,300/500,'high');
%截止频率300Hz,采样频率1000Hz的9阶巴特沃思高通滤波器
[b,a] = cheby1(9,0.5,300/500);
%截止频率300Hz,采样频率1000Hz,通带容限(Rp)0.5dB的9阶契比雪夫低通滤波器
[b,a] = cheby2(9,20,300/500);
%截止频率300Hz,采样频率1000Hz,阻带容限(Rs)20dB的9阶契比雪夫低通滤波器
[b,a] = ellip(3,1,60,[0.4 0.7],'stop');
%截止频率0.4,0.7,通带容限(Rp)1dB,阻带容限(Rs)60dB的3阶椭圆带阻滤波器
[b,a] = butter(5,0.4,'s');%截止频率0.4的5阶模拟巴特沃思低通滤波器
```

表 10 3

滤波器类型	设计函数
巴特沃思	[n,Wn] = buttord(Wp,Ws,Rp,Rs)
契比雪夫 I 型	[n,Wn] = cheb1ord(Wp, Ws, Rp, Rs)
契比雪夫 II 型	[n,Wn] = cheb2ord(Wp, Ws, Rp, Rs)
椭圆	[n,Wn] = ellipord(Wp, Ws, Rp, Rs)

(2)根据频域指标设计 IIR 滤波器

表 10-3 列出的函数可以设计出满足频域设计指标的最低阶滤波器，这些函数在设计复合型滤波器时是很方便的。注意表中函数返回的是滤波器的阶数和截止频率。设计模拟滤波器加‘s’。

【例 10.3.1-1】 设采样频率 1000Hz。设计一低通滤波器，在 0～100 Hz 范围，通带容限小于 3 dB；150 Hz 到奈氏频率，幅值衰减大于 15dB。

```
Wp = 100/500; Ws = 150/500;
Rp   3; Rs = 15;
[n,Wn] = cheb2ord(Wp,Ws,Rp,Rs)
```

结果是

```
ans=
n = 3
Wn = 0.2609
```

构造滤波器

```
[b,a] = cheby2(n,Rs,Wn);
```

【例 10.3.1-2】 设采样频率 10kHz，设计带通滤波器，通频带 1～2 k Hz，通带容限 1 dB，阻带上下限频率分别是 500 Hz，2.5kHz，阻带衰减不小于 60dB。

```
[n,Wn] = buttord([1000 2000]/5000,[500 2500]/5000,1,60)
```

结果是

```
ans=
n = 12
Wn = 0.1951 0.4080
[b,a] = butter(n,Wn);
```

(3)模拟 IIR 滤波器比较

a. 巴特沃思滤波器

它是理想低通滤波器在 $\Omega=0$ 和 $\Omega=\infty$ 时响应的最佳泰勒逼近。在区间($\Omega=0,\Omega=\infty$)，幅值平方响应单调下降。对任意阶次 N，在 $\Omega=0$ 和 $\Omega=\infty$ 处，幅值平方

响应的导数(一阶、二阶…$N-1$阶)皆为零,故巴特沃思滤波器又称'最平'的幅频响应滤波器。

$\Omega=1$ 时,$|H(j\Omega)|=0.707$。

b. 契比雪夫Ⅰ型滤波器

整个通频带内,理想的和实际的频率响应具有最小的误差,但通带有Rp分贝的等幅纹波。从通带到阻带的过渡过程比巴特沃思滤波器小得多。阻带频率响应很均匀。

$\Omega=1$ 时,$|H(j\Omega)|=10^{-Rp/20}$。

c. 契比雪夫Ⅱ型滤波器

整个阻带内,理想的和实际的频率响应具有最小的误差,阻带等幅衰减Rs分贝。通带频率响应很均匀,无纹波,过渡过程短。

$\Omega=1$ 时,$|H(j\Omega)|=10^{-Rs/20}$。

d. 椭圆滤波器

通带、阻带都有等幅纹波,可以构造具有最低阶数的滤波器。在同样的技术指标下,椭圆滤波器过渡过程最短。

$\Omega=1$ 时,$|H(j\Omega)|=10^{-Rp/20}$。

e. 贝塞尔滤波器

模拟贝塞尔滤波器在零频率处有最均匀群时延,且在整个通带内,群时延近似常数,因此滤波后的信号仍保持原来波形。若要求阻带具有同样衰减,则需较其他滤波器有更高阶次。

$\Omega=1$ 时,$|H(j\Omega)|<0.707$,且随阶次 N 增大而减小。

2. 直接IIR滤波器设计

直接IIR滤波器设计是指在离散域直接设计滤波器。与模拟原型方法不同,直接设计不受标准低通、高通、带通、带阻等限制,可以根据任何指定频率的响应(也可以是多带的)设计滤波器。下面讨论yulewalk函数。

yulewalk根据指定频率响应设计IIR数字滤波器。yulewalk的名字本身就反映了它是一种求滤波器分母系数的方法:对指定频率响应作逆FFT,用得出的相关系数求解修正的Yule-Walker方程,求出滤波器分母系数。语句:

```
[b,a] = yulewalk(n,f,m)
```

返回滤波器分子、分母的系数,滤波器幅频特性近似与参数f,m匹配。f是区间为[0,1]的频率向量,单位是奈氏频率,f的第一个元素必须是0,最后一个元素是1,其他频率点按升序排列,允许同一频率点出现两次。m是包含所有指定频率点上期望幅值响应的向量,m必须与f同样大小。f和m可以表示任何分段线性的幅值响应以及多带幅值响应。

【例 10.3.1-3】 用 yulewalk 设计一多带滤波器，并绘制其幅值响应曲线。

```
m = [0 0 1 1 0 0 1 1 0 0];
f = [0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 1];
[b,a] = yulewalk(10,f,m);
[h,w] = freqz(b,a,128);
plot(f,m,w/pi,abs(h))
```

图 10.3.1-1 是幅值响应曲线。

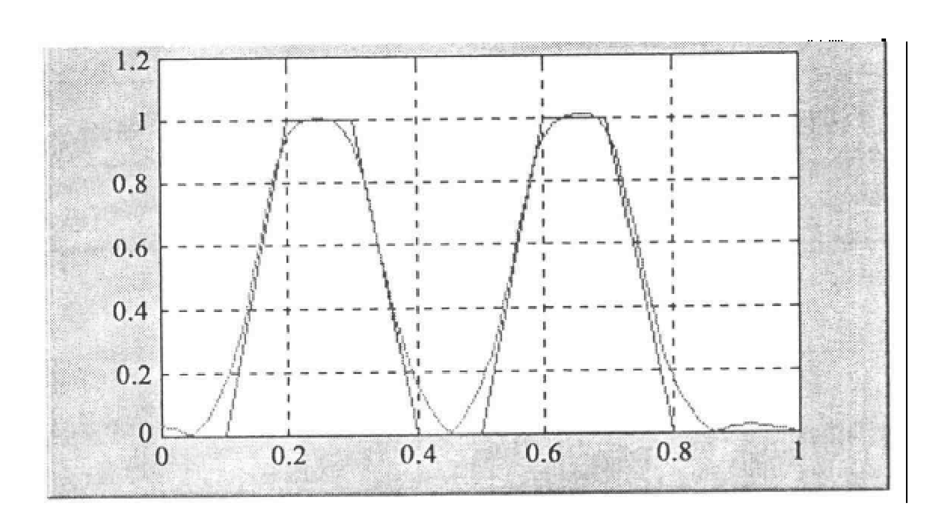


图 10.3.1-1　幅值响应曲线

10.3.2 FIR 滤波器

由式(10.2.2-1)我们知道，FIR 滤波器只有零点，若要获得与 IIR 滤波器同样的通带和阻带衰减特性，需要更高的阶次，延时也更大。但 FIR 滤波器也有突出的优点，如系统总是稳定的，易实现线性相位，容易设计多通带(多阻带)滤波器。

1. 设计方法

表 10-4 是 MATLAB 设计 FIR 滤波器的方法及所用函数。使用缺省值时，函数 fir1，fir2，firls，remez，fircls，fircls1 和 firrcos 构造 Ⅰ、Ⅱ 型线性相位 FIR 滤波器。firls 和 remez 构造 Ⅲ、Ⅳ 型线性相位 FIR 滤波器。cremez 可以设计线性或非线性相位 FIR 滤波器。

注意，由于 Ⅱ 型滤波器在高频段的频率响应为零，fir1 不设计 Ⅱ 型高通和带阻滤波器。当 n 是奇数且带参数'stop'或'high'时，fir1 返回 Ⅰ 型滤波器，并使阶次加 1。

2. 线性相位滤波器

线性相位滤波器系数具有奇对称或偶对称关系。由于这种关系，线性相位滤波

器对频率的响应有一定的限制。表10-5是各种线性相位滤波器频响特性。

表10-4

方　法	说　明	函数讲法
窗函数法	应用窗函数截短数据序列	b=fir1(n,Wn,options) b=fir2(n,f,m,options)
带瞬态段的多带法	最小方差逼近	b=firls(n,f,m,options) b=remez(n,f,m,options)
约束最小平方法	最小化整体方差	fircls,fircls1
频率响应	非线性相位和复合滤波器	cremez

表10-5

滤波器类型	阶次 n	系数对称性	响应 $H(f)$ (f 是奈氏频率)	
			$f=0$	$f=1$
Ⅰ	偶数	偶数： $b(k)=b(n+2-k)\ k=1,\cdots,n+1$	无限制	无限制
Ⅱ	奇数		无限制	$H(1)=0$
Ⅲ	偶数	奇数： $b(k)=-b(n+2-k)\ k=1,\cdots,n+1$	$H(0)=0$	$H(1)=0$
Ⅳ	奇数		$H(0)=0$	无限制

整个频率段上，线性相位滤波器的相延迟和群延迟相等且为常数。一个 n 阶线性相位滤波器，其群延迟为 $n/2$，滤波后的信号只是被延迟 $n/2$ 个时间步长，其傅氏变换幅值与滤波器频响成比例。因此，经线性相位滤波器滤波后的信号无相位失真。

3. 窗函数设计法

对于理想的低通数字滤波器频率特性，其幅频特性 $|H(e^{j\omega})|=1$，相频特性 $\phi(\omega)=0$。滤波器脉冲响应

$$h(n)=\frac{1}{2\pi}\int_{-\pi}^{\pi}H(e^{j\omega})e^{j\omega}d\omega=\frac{1}{2\pi}\int_{-\omega_0}^{\omega_0}e^{j\omega}d\omega=\frac{\omega_0}{\pi}\mathrm{sinc}(\frac{\omega_0}{\pi}n) \qquad (10.3.2\text{-}1)$$

这是非因果系统，在物理上是不可实现的。但如果把 $h(n)$ 截短，取其中间部分，此时

$$h(n) = h_d(n - M/2) \quad n = 0,1,\cdots,M \qquad (10.3.2\text{-}2)$$

成为因果的、长度是 $M+1$ 的有限长序列，这就是线性相位 FIR 滤波器。

可以用 sinc 构造一个长度 51，截止频率 0.4rad/s 的低通滤波器。

```
b = 0.4 * sinc(0.4 * (-25:25));
```

这是在 $h(n)$ 上加矩形窗的结果。下面语句画出滤波器的频率响应(图 10.3.2-1)。

```
[H,w] = freqz(b,1,512,2);
plot(w,abs(H)), grid
```

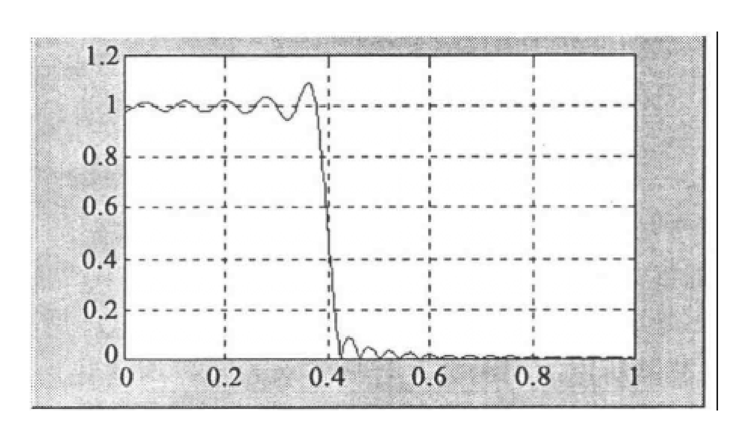


图 10.3.2-1

从图中可以看到，通带、阻带都有纹波，越靠近截止频率，幅度越大，这称作吉布斯现象。吉布斯现象是由于 $h(n)$ 被截短的结果，对无穷序列 $h(n)$ 仅取其中长 0～M的序列，等于在 $h(n)$上施加了长为 $M+1$ 的矩形窗。窗函数的旁瓣是产生吉布斯现象的原因。为了减弱吉布斯现象，可以采用旁瓣小的窗函数。下面的语句改用海明窗：

```
b = b.* hamming(51)';
[H,w] = freqz(b,1,512,2);
plot(w,abs(H)),grid
```

图 10.3.2-2 是加海明窗的频率响应。从图中已看不到纹波，但过渡过程要大些。fir1 和 fir2 都是基于窗函数法的，给定滤波器参数，函数返回滤波器的系数。

(1)fir1

fir1 是设计线性相位滤波器的经典方法。语句

```
n = 50;
Wn = 0.4;
b = fir1(n,Wn);
```

构造 n 阶滤波器，返回滤波器系数。参数 n，Wn 的意义同前，如果 Wn 是多元素向

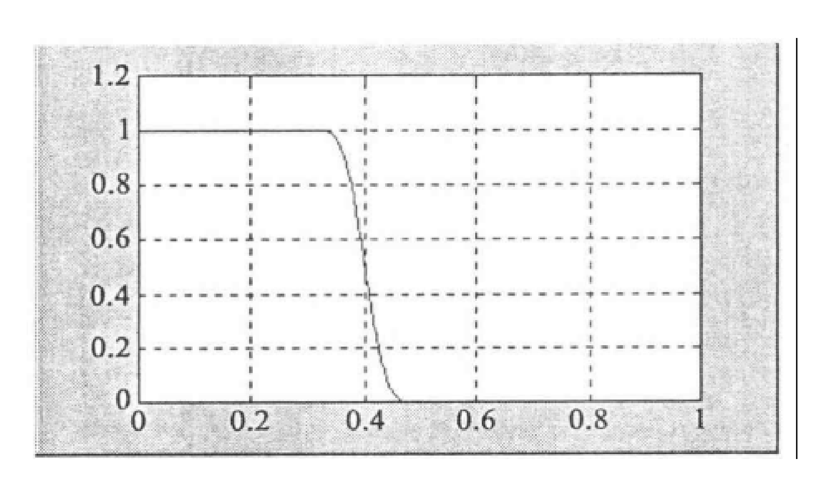


图 10.3.2-2

量,Wn = [w1 w2 ... wn],fir1 返回 n 阶多带滤波器,通带频率

0 <ω< w1, w1 <ω< w2, ..., wn <ω< 1.

在 fir1 中附加参数'DC-0'指定第一频段是阻带,'DC-1'则是通带。设计高通和带阻的方法见 IIR 滤波器。fir1 还有一可选的 windows 参数,是长度为 n+1 的列向量,表示构造滤波器所用的窗函数。若缺省 windows,fir1 用海明窗。

(2)fir2

fir2 也是构造加窗的 FIR 滤波器,与 fir1 不同的是,其构造的滤波器具有任意形状的分段线性频响。程序段

```
n = 50;
f = [0 .4 .5 1];
m = [1 1 0 0];
b = fir2(n,f,m);
```

构造 n 阶 FIR 滤波器,滤波器的幅频特性由参数 f,m 指定。与 fir2 对应的 IIR 滤波器构造函数是 yulewalk。f,m 的意义及定义限制可参考 yulewalk。

4. 带过渡段的多带滤波器(firls 和 remez)

firls 和 remez 设计滤波器的方法更灵活通用,定义参数的自由度较 fir1,fir2 大,允许包含过渡段或那些并未对误差最小化的频段。多段滤波器的各频段的性能取决于其误差最小化的权。

firls 算法的思想是使期望频响与实际频响的整体方差最小。remez 则采用 Parks-McClellan 算法,即用 Remez 交换算法和契比雪夫逼近理论设计滤波器,使实际的和期望的频率响应间达到最佳适配,使两者间的误差最小,此即最优滤波器,也称最小滤波器。remez 设计的滤波器在频率响应中出现等幅纹波,因此亦称等纹波滤波器。

(1)基本结构 调用 firls 和 remez 的语法一样,两者只是采用的算法不同。用缺省方式时,两者设计 I 型或 II 型滤波器。

【例 10.3.2-1】 分别用 firls 和 remez 设计一个在 0~0.4(奈氏频率单位)间的频响为 1,0.45~1 间的幅值为 0 的低通滤波器。比较两者频响曲线。

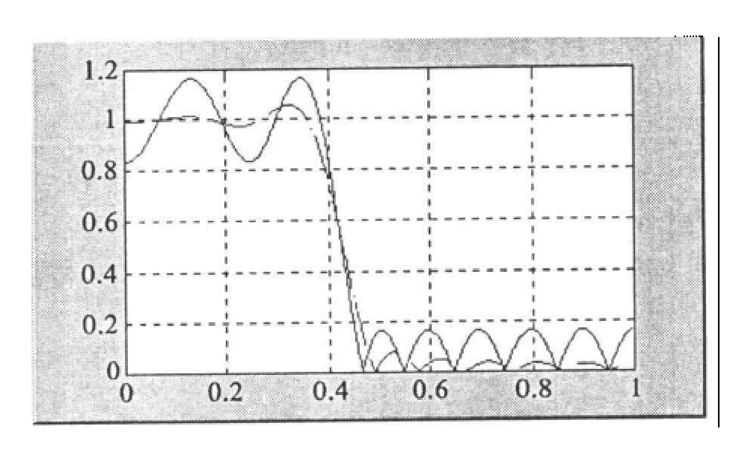


图 10.3.2-3

```
n = 20; % filter order
f = [0 0.4 0.45 1]; % frequency band edges
a = [1 1 0 0]; % desired amplitudes
b = remez(n,f,a);   %designing by remez
bb = remez(n,f,a);   %designing by firls
[h,w]=freqz(b);
[h1,w]=freqz(bb);
plot(w/pi,abs(h),w/pi,abs(h1),'--')
grid
```

图 10.3.2-3 是两种函数构造的滤波器的幅频响应,有等幅纹波的是 remez 的频率响应曲线。可以看出 firls 设计的滤波器的响应比 remez 的好,但在过渡带 0.4~0.5处,remez 滤波器更接近理想值,说明 remez 滤波器在这一带的最大误差更小。实际上,无论是过渡段,还是整个滤波频段,remez 滤波器的误差可能都是最小的。

firls 和 remez 的算法考虑较短频率段为直线。基于这种方法,可以构造任何带有过渡段的分段线性滤波器。下面构造一个带通滤波器。

```
f = [0 0.3 0.4 0.7 0.8 1]; % band edges in pairs
a = [0 0 1 1 0 0]; % bandpass filter amplitude
```

向量 f 和 a 定义了 5 个频段:

- 两个阻带 0.0~0.3,0.8~1.0

●一个通带　　0.4～0.7

●两个过渡带　0.3～0.4,0.7～0.8

构造通带和阻带滤波器的向量：

```
f = [0 0.7 0.8 1]; % band edges in pairs
a = [0 0 1 1]; % highpass filter amplitude
f = [0 0.3 0.4 0.5 0.8 1]; % band edges in pairs
a = [ 1 1 0 0 1 1]; % bandstop filter amplitude
```

构造多带带通滤波器的向量：

```
f = [0 0.1 0.15 0.25 0.3 0.4 0.45 0.55 0.6 0.7 0.75 0.85 0.9 1];
a = [1 1 0 0 1 1 0 0 1 1 0 0 1 1];
```

下面的向量可以构造将线性连接的通带和阻带作为过渡带的滤波器，这样可以控制宽过渡区的'失控'频响。

```
f = [0 0.4 0.42 0.48 0.5 1];
a = [1 1 0.8 0.2 0 0]; % passband,linear transition,stopband
```

(2)加权算法　firls 和 remez 可以对不同的频段设置不同的权，表示对不同频段误差最小化的程度。权向量的定义方法见【例 10.3.2-2】。

【例 10.3.2-2】　设计一低通滤波器，其阻带容限是通带的 1/10。

```
n = 20; % filter order
f = [0 0.4 0.5 1]; % frequency band edges
a = [1 1 0 0]; % desired amplitudes
w = [1 10]; % weight vector
b = remez(n,f,a,w);
```

注意，权向量 w 的长度是 f 的一半，每个频段有一个权。

(3)反对称滤波器 Hilbert　变换器调用时加入字符串参数'h'或'Hilbert'时，firls 和 remez 构造奇对称 FIR 滤波器，即Ⅲ型(偶数阶)或Ⅳ型(奇数阶)滤波器。理想的 Hilbert 变换器具有反对称性质，且在整个频段的频率响应都是 1。

【例 10.3.2-3】　试设计高通和带通 Hilbert 变换器，并绘制频响曲线。

```
b = remez(21,[0.05 1],[1 1],'h'); % highpass Hilbert
bb = remez(20,[0.05 0.95],[1 1],'h'); % bandpass Hilbert
[h w]=freqz(b);
[h1 w]=freqz(bb);
plot(w/pi,abs(h))
grid
plot(w/pi,abs(h1))
 grid
```

图 10.3.2-4(a)、图 10.3.2-4(b)分别是高通、带通 Hilbert 的频率响应。

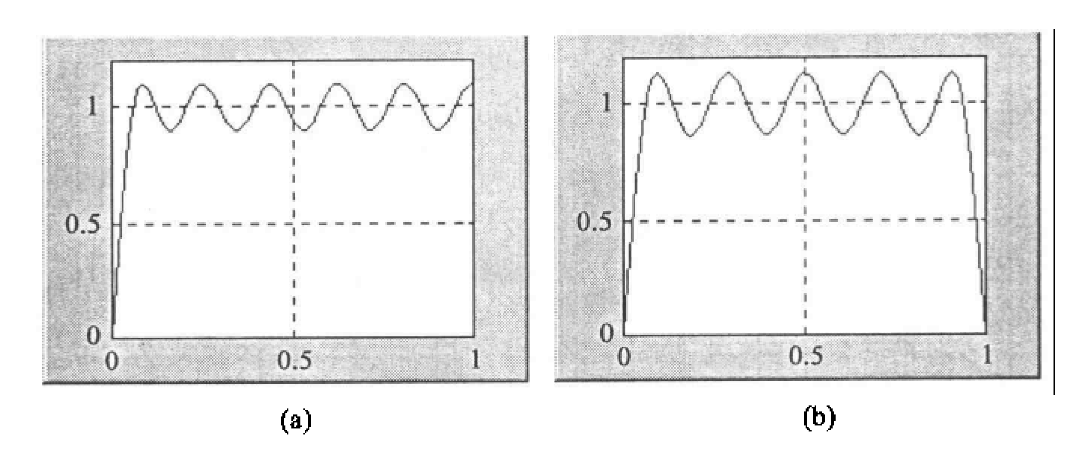


图 10. 3. 2-4

将信号 x 通过这些滤波器，可以得到延迟的 Hilbert 变换。

```
Fs = 1000; % sampling frequency
t = (0:1/Fs:2)'; % two second time vector
x = sin(2 * pi * 300 * t); % 300 Hz sine wave example signal
xh = filter(bb,1,x); % Hilbert transform of x
```

对应于 x 的解析信号是复数信号，x 是实部，x 的 Hilbert 变换是虚部。对于 FIR 方法，必须将 x 延迟 $n/2$(n 是滤波器阶次)，以产生解析信号。

```
xd = [zeros(10,1); x(1:length(x)-10)]; % delay 10 samples
xa = xd + j * xh; % analytic signal
```

对于奇数阶滤波器不能直接应用此法，需先进行非整延迟，可以用 hilbert 函数估计解析信号，或用 resample 函数对采样信号作非整数采样延迟。

(4)微分器　根据傅氏变换的性质，时域信号的微分等于该信号的傅氏变换乘以一个虚数因子。因此，要对信号微分，可使其通过一个响应为 $H(\omega)=j\omega$ 的滤波器。可以通过在调用 firls 和 remez 时，加入字串参数 d 或 differentiator 模仿微分器(带延迟)。如语句

```
b = remez(21,[0 1],[0 pi * Fs],'d');
```

为了得到正确的导数，可用 pi * Fs 进行标定(Fs 是以 Hz 为单位的采样频率)。对于Ⅲ型滤波器，微分频段必须在奈氏频率前结束，并要调整幅值向量，以保证正确的斜率。

```
bb = remez(20,[0 0.9],[0 0.9 * pi * Fs],'d');
```

在 d 模式下，remez 在非零值段用 $1/\omega$ 对误差加权来最小化最大相关误差。firls 在非零值段用 $1/\omega^2$ 对误差加权。下面绘制 b，bb 的图形。

```
[h w]=freqz(b);
```

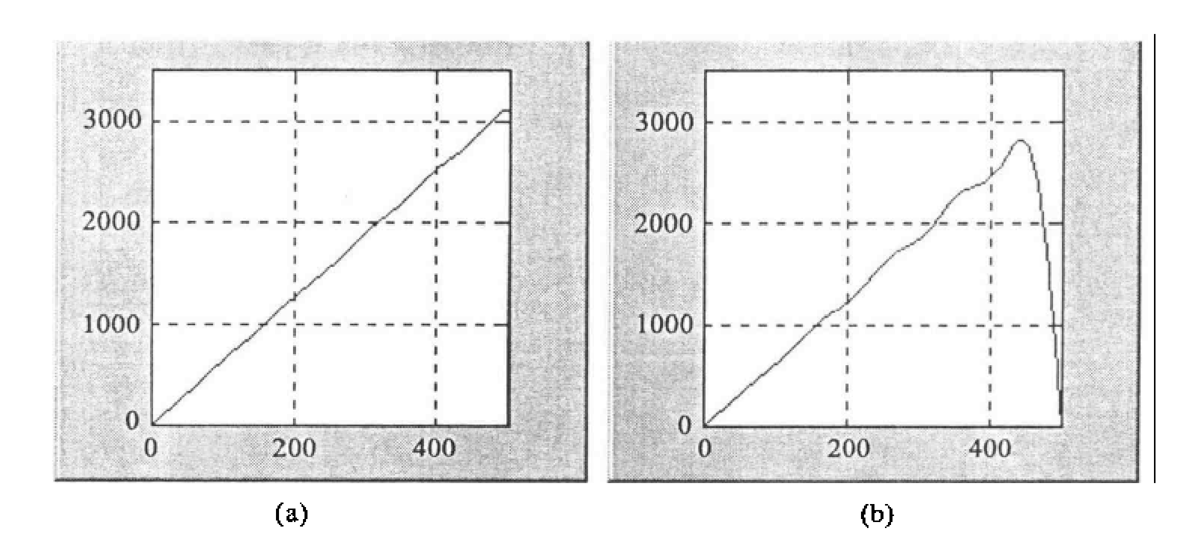


图 10.3.2-5

```
[h1 w]=freqz(bb);
plot(w/pi,abs(h))
grid
plot(w/pi,abs(h1))
    grid
```

图 10.3.2-5(a)、(b)是 b,bb 的频率响应。

5. 约束最小平方(CLS)FIR 滤波器

设计约束最小平方 FIR 滤波器(CLS)的特点是无需明确给定过渡带的频响幅值,这种性质在某些场合是很有用的。如果信号与噪声混杂在同一频率中,它们之间的过渡带不是那么清楚的,若只考虑如何控制吉布斯现象,这时滤波器的设计一般不指定过渡带,而把截止频率、通带或阻带的边沿作为参数。CLS 法的特点是可以指定幅值响应纹波的上、下限,在整个频率范围使误差平方最小化。用于 CLS 法的函数是 fircls,fircls1。

fircls 调用语法:

```
b = fircls(n,f,amp,up,lo)
fircls(n,f,amp,up,lo,'design_flag')
```

参数说明:f 为转折频率向量,第一个是 0,最后是 1,其他按升序排列;amp 为各频段的频响期望值,长度 f 为 1;up,lo 为通带及阻带容限;'design_flag'可以是 trace,即用文字表示误差;plots 为绘制频率响应,并放大纹波图形;both 表示可以同时使用 trace,plots。

fircls1 调用语法:

```
b = fircls1(n,wo,dp,ds,wt)
b = fircls1(n,wo,dp,ds,wt,'high')
b = fircls1(n,wo,dp,ds,wp,ws,k)
b = fircls1(n,wo,dp,ds,wp,ws,k,'high')
b = fircls1(n,wo,dp,ds,'design_flag')
```

参数说明：wo 为转折频率；dp，ds 同 up，lo；'high'，'design_flag' 意义见 fircls；wt 表示 wt 上的频响满足设计要求；wp 表示 L2 权函数通带频率；ws 为阻带频率；k 表示通带 L2 误差与阻带 L2 误差比。

(1)低通、高通 CLS 滤波器

【例 10.3.2-4】 设计一个 61 阶 CLS 滤波器，截止频率 0.3，通带容限0.02，阻带 0.008。

为达到设计要求，用函数 fircls1。下面是实现的程序段。

```
n = 61;
wo = 0.3;
dp = 0.02;
ds = 0.008;
h = fircls1(n,wo,dp,ds,'plot');
```

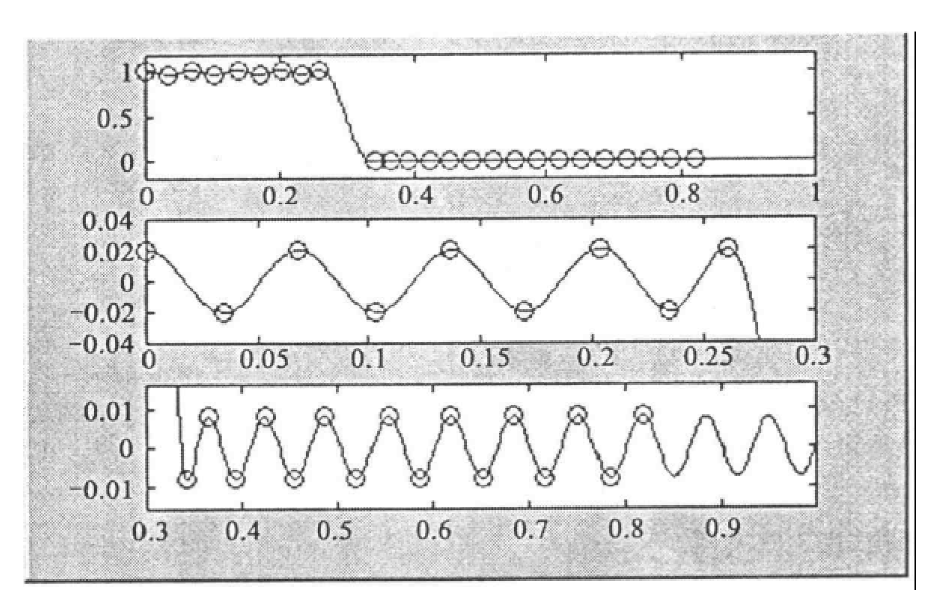


图 10.3.2-6

图 10.3.2-6 是滤波器频率响应。

(2)多带 CLS 滤波器

可以用与 FIR 滤波器同样的方法设计 CLS 滤波器。

【例 10.3.2-5】 设计一个129阶CLS滤波器,各频段参数如下:

0~0.3:频响幅值0,纹波上界0.005,下界-0.005。

0.3~0.5:频响幅值0.5,纹波上界0.51,下界0.49。

0.5~0.7:频响幅值0,纹波上界0.03,下界-0.03。

0.7~0.9:频响幅值1,纹波上界1.02,下界0.98。

0.9~1:频响幅值0,纹波上界0.05,下界-0.05。

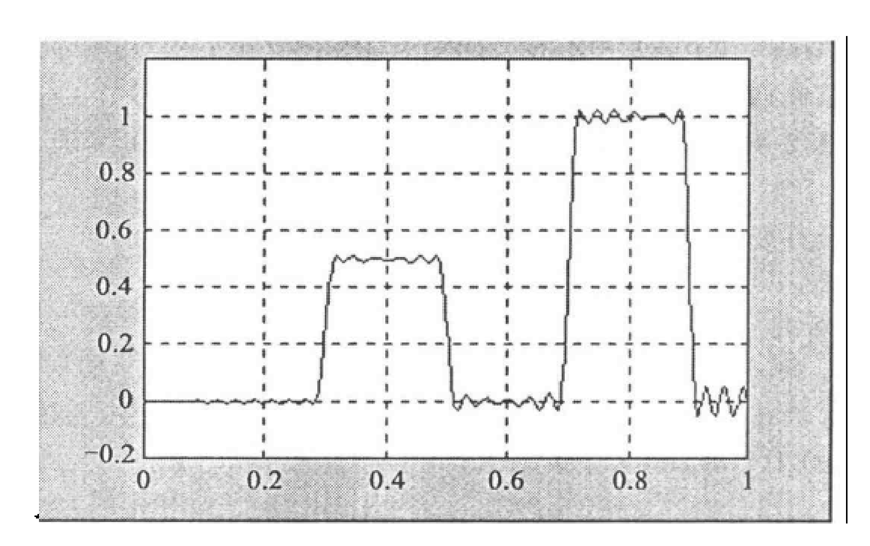


图 10.3.2-7

输入命令:

```
n = 129;
f = [0 0.3 0.5 0.7 0.9 1];
a = [0 0.5 0 1 0];
up = [0.005 0.51 0.03 1.02 0.05];
lo = [-0.005 0.49 -0.03 0.98 -0.05];
h = fircls(n,f,a,up,lo,'plot');
```

图 10.3.2-7 是滤波器频率响应。

(3)加权CLS滤波器

fircls1 可以根据各频段误差最小化的权设计FIR滤波器。

【例 10.3.2-6】 设计一个55阶FIR滤波器,转折频率wo=0.3,dp=0.02,ds=0.004,权函数通带边沿频率wp=0.28,阻带边沿频率ws=0.32,比率k=10(图10.3.2-8)。

输入命令:

```
n = 55;
```

```
wo = 0.3;
dp = 0.02;
ds = 0.004;
wp = 0.28;
ws = 0.32;
k = 10;
h = fircls1(n,wo,dp,ds,wp,ws,k,'plot');
```

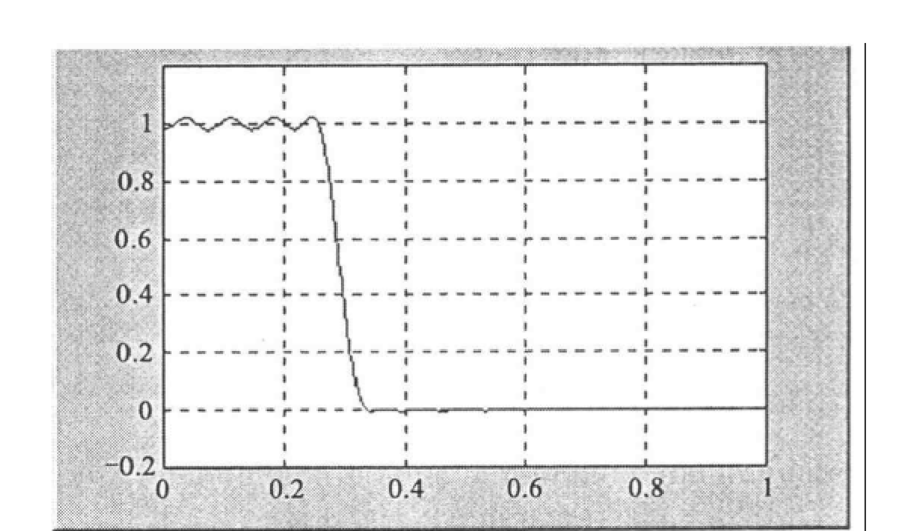


图 10.3.2-8

10.4　随机信号处理

工程实际中，随机信号处理有着广泛的应用，如信号检测、系统辨识、雷达、声纳、机器故障诊断等领域。MATLAB 也有处理随机信号的函数，如估计离散信号的互相关、协方差及谱密度等。.

10.4.1　互相关和协方差

函数 xccor 和 xcov 估计随机过程的互相关和互协方差序列(特例是自相关、自协方差)。

随机信号互相关函数定义：

$$R_{xy}(m) = E[x*(n)y(n+m)] \tag{10.4.1-1}$$

互协方差：

$$C_{xy}(m) = E\{[x(n)-m_x]*[y(n+m)-m_y]\} \tag{10.4.1-2}$$

式中 $x(n)$，$y(n)$是平稳随机过程，m_x，m_y 是两个序列的均值。由于样本只是整个随机过程的有限长数据，因此只能对随机过程的数字特征进行估计。

1. xcorr(互相关估计)

语法：

```
c = xcorr(x,y)
c = xcorr(x,y,'option')
```

当 $x(n)$，$y(n)$的长度是 M 时，xcorr 估计出长度为 $2M-1$ 的互相关序列，缺省参数 option 时，计算非归一化的行相关。

$$\hat{R}_{xy}(m) = \sum_{n=0}^{M-|m|-1} x*(n)y(n+m) \tag{10.4.1-3}$$

输入：

```
x=2:6;
y=4:8;
c=xcorr(x,y)
```

结果是

```
c=
16.0000    32.0000    47.0000    28.0000    12.0000
```

若 option＝biased，xcorr 计算互相关的有偏估计

$$\hat{R}_{xy,biased}(m) = \frac{1}{N}\hat{R}_{xy}(m) \tag{10.4.1-4}$$

若 option＝unbiased，xcorr 计算互相关的无偏估计

$$\hat{R}_{xy,unbised}(m) = \frac{1}{N-|m|}\hat{R}_{xy}(m) \tag{10.4.1-5}$$

若 option＝coeff，xcorr 对序列进行归一化处理，使零滞后的自相关序列为 1。计算互相关的无偏估计。

2. xcov(互协方差估计)

xcov 估计随机过程的互协方差，互协方差是除去均值的互相关序列。

语法：

```
v = xcov(x,y)
v = xcov(x,y,'option')
```

参数意义同 xcorr。

计算 x，y 的互协方差

```
   c=xcov(x,y)
c =
  -1    0    2    0    -1
```

3. 处理多通道信号

对多通道信号，xcorr 和 xcov 同时估计所有通道的互相关和互协方差。若 X 是一个 M×N 的信号矩阵，有 N 个通道，则 xcorr(X) 返回一个 $(2M-1)\times N^2$ 矩阵，其 N^2 列是 N 个通道的自相关和互相关序列。如有三通道信号矩阵 X：

```
X=[x1,x2,x3];
R=xcorr(X);
```

R 的结果按如下形式排列：

R=[Rs1s1 Rs1s2 Rs1s3 Rs2s1 Rs2s2 Rs2s3 Rs3s1 Rs3s2 Rs3s3]

另有两个函数 cov 和 corrcoef 计算不同通道间的协方差和相关系数矩阵。

10.4.2 谱 分 析

随机过程的基本数字特征是其数学期望和相关函数或频域内的功率谱密度。工程中遇到的实际问题，只要满足平稳性的要求，我们就可以通过研究它（它们）的相关性，进行谱分析。谱分析是检测随机信号的有效方法。

随机序列的自功率谱密度(PSD)：

$$P_{xx}(\omega)=\sum_{-\infty}^{\infty}R_{xx}(m)e^{-j\omega m} \tag{10.4.2-1}$$

互功率谱密度(CSD)：

$$P_{xy}(\omega)=\sum_{-\infty}^{\infty}R_{xy}(m)e^{-j\omega m} \tag{10.4.2-2}$$

功率谱表示信号的功率在频域随频率的分布情况。经典的谱估计方法在 MATLAB 中主要是直接法、Welch 法等，可以用 psd，csd，tfe 和 cohere 等函数实现。参数模型谱估计，有最大熵谱估计函数 lpc 及其他相关函数。

1. 直接法(周期图法)

随机信号 $x(n)$ 的谱估计是对其 N 点观察数据 $x_N(n)$ 进行傅氏变换，得到 $X_N(e^{j\omega})$，然后取 $X_N(e^{j\omega})$ 幅值的平方，再除以 N。

$$\hat{P}(\omega)=\frac{1}{N}|X_N(\omega)|^2 \tag{10.4.2-3}$$

【例 10.4.2-1】 信号 $x(n)$ 包含两个正弦信号和噪声，取 1001 个数据，估计其谱密度。

```
Fs = 1000; % sampling frequency
t = 0:1/Fs:1; % one second worth of samples
xn = sin(2*pi*50*t) + 2*sin(2*pi*120*t) + andn(size(t));
Pxx = abs(fft(xn,1024)).^2/1001;
```

直接法取无限序列 $x(n)$ 的 N 个数据，实际是对 $x(n)$ 加了一个长度为 N 的矩

型窗。对于定长度 N，直接估计是渐进有偏估计，但它是渐进无偏的。采样点 N 愈多，估计值愈接近真值，但是不相关的点数也随之增多，加剧了图形起伏变化。下面的例子说明了这点。

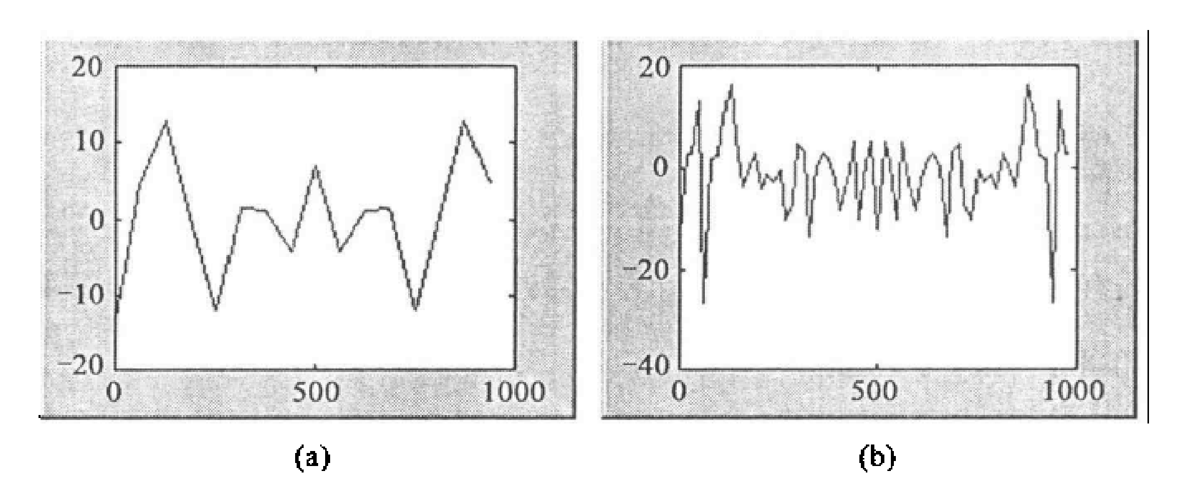


图 10.4.2-1

```
Pxx_short = abs(fft(xn,16)).^2/16;
Pxx= abs(fft(xn,64)).^2/64;
plot((0:15)/16*Fs,10*log10(Pxx_short))
plot((0:63)/64*Fs,10*log10(Pxx))
```

图 10.4.2-1(a)、(b)是不同数据点数的谱估计图形。

2. Bartlett 法

通过对 $x_N(n)$ 分段，分别求各段的功率谱，然后求和、平均，使当 N 增加时，可以减小方差，使曲线平滑。下面是 Bartlett 法的例子(图 10.4.2-2)。

```
Pxx=(abs(fft(xn(1:256))).^2+abs(fft(xn(257:512))).^2+abs(fft(xn(513:
768))).^2)/(256*3);
plot((0:255)/256*Fs,10*log10(Pxx))
```

3. Welch 法(加权交叠平均法)

Welch 法是对 Bartlett 的改进，Welch 法对 $x_N(n)$ 分段时使各段数据部分重叠。另外，每段数据的窗口也不一定是矩形窗，可以是海明窗、汉宁窗等，这样可以改善由矩形窗旁瓣引起的谐波失真。

下面是 6 段重叠平均的例子(图 10.4.2-3)：

```
Pxx=(abs(fft(xn(1:256))).^2+abs(fft(xn(129:384))).^2+abs(fft(xn(257:
512))).^2+abs(fft(xn(385:640))).^2+
abs(fft(xn(513:768))).^2 + abs(fft(xn(641:896))).^2)/(256*6);
plot((0:255)/256*Fs,10*log10(Pxx))
```

比较图 10.4.2-2 和图 10.4.2-3，可以看出 Welch 的图形较为平滑。

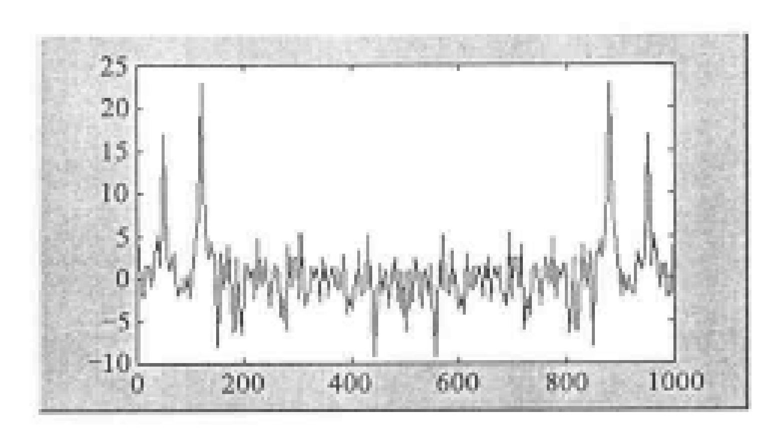


图 10.4.2-2

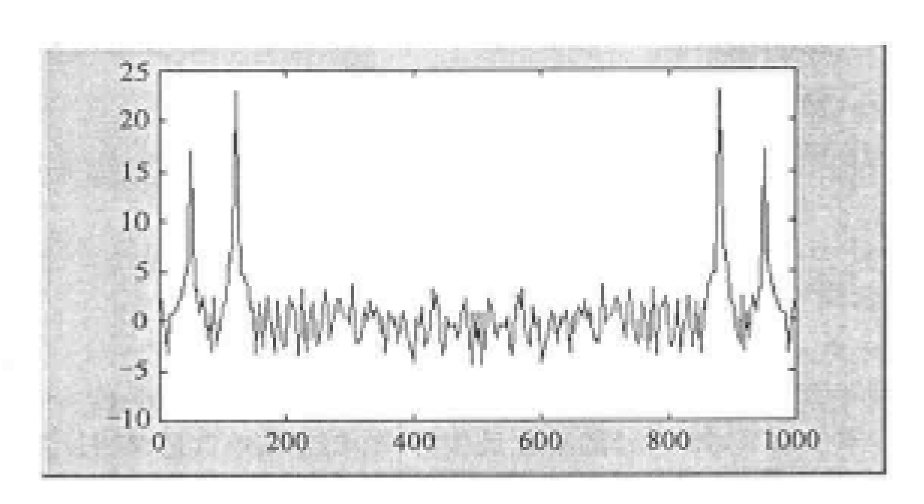


图 10.4.2-3

下面是用汉宁窗的例子(图 10.4.2-4)：

```
w = hanning(256)';
Pxx = ( abs(fft(w. * xn( 1:256))). ^ 2 +abs(fft(w. * xn(129:384))). ^ 2 +
abs(fft(w. * xn(257:512))). ^ 2 + abs(fft(w. * xn(385:640))). ^ 2 +
abs(fft(w. * xn(513:768))). ^ 2+abs(fft(w. * xn(641:896))). ^ 2)/(norm(w) ^ 2 *
6);
plot((0:255)/256 * Fs,10 * log10(Pxx))
```

图 10.4.2-4 的谱峰值间隔最大，曲线变化是几种估计中最平滑的。

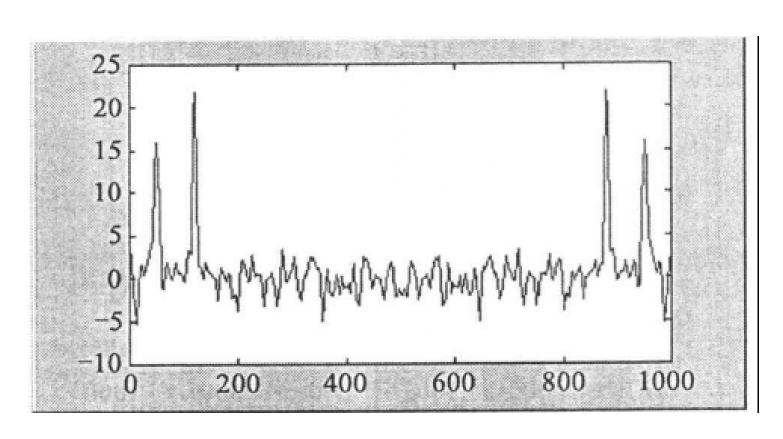


图 10.4.2-4

10.4.3 谱分析函数

1. pwelch(功率谱密度估计函数)

pwelch 用 Welch 法估计功率谱。

语法:

```
[Pxx,Pxxc,f] = pwelch(x,nfft,Fs,window,noverlap,p)
```

缺省参数调用

```
Pxx = pwelch(xn);
```

此时 nfft=256,window=256(长 256 的汉宁窗),重叠量 noverlap=0。Pxx 的单位是功率/单位频率,如 $x(n)$ 的单位是伏特,则 Pxx 的单位是 W/Hz。

仍用前面的例子,设重叠量为 128。

```
nfft = 256; % length of FFT
window = hanning(256); % window function
noverlap = 128; % number of samples overlap
Pxx = pwelch(xn,nfft,Fs,window,noverlap);
```

Pxx 用采样频率的倒数规格化,无输出参数的 pwelch 绘制 0-Fs/2 频率范围的 Pxx 图形。

```
pwelch(xn,nfft,Fs,window,noverlap)
```

图 10.4.3-1 是输出图形。若想另画图形,可以增加输出频率向量的参数:

```
[Pxx,f] = pwelch(xn,nfft,Fs,window,noverlap);
plot(f,10 * log10(Pxx))
```

2. csd(估计互谱密度)

csd 是复函数,用 Welch 法估计同长序列 x(n)和 y(n)的互谱密度。两个序列的分段和加窗同 pwelch。

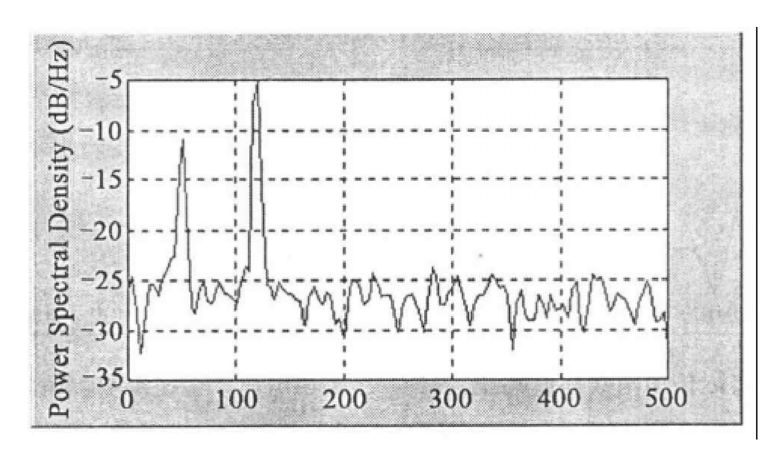


图 10.4.3-1

语法：

```
[Pxy,Pxyc,f]=csd(x,y,nfft,Fs,window,noverlap,p)
```

3. 区间估计

pwelch 和 cds 的参数 p 是置信度系数，值域(0,1)。假设函数计算结果服从 X^2 分布，对区间[Pxx－Pxxc(:,1) Pxx＋Pxxc(:,2)]的置信度是 p * 100%。注意，作区间估计时，overlap＝0，否则结果不可靠，此时函数给出出错信息。

4. 传递函数估计

Welch 法还可用于非参数化的系统辨识。设 H 是一个 LTI 系统，$x(n)$，$y(n)$ 是 H 的输入和输出。那么 $x(n)$的 PSD 与 $x(n)$和 $y(n)$的 CSD 有如下关系

$$P_{xy}(\omega) = H(\omega)P_{xx}(\omega) \tag{10.4.3-1}$$

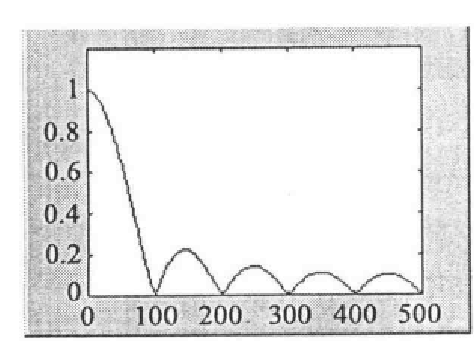
H 的传递函数估计

$$\hat{H}(\omega) = \frac{\hat{P}_{xy}(\omega)}{\hat{P}_{xx}(\omega)} \tag{10.4.3-2}$$

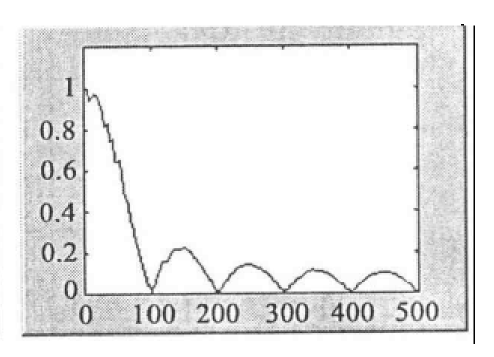
函数 tfe 用 Welch 法计算 PSD 和 CSD，然后用式(10.4.2-3)估计传递函数。tfe 的语法和 csd 一样。

【例 10.4.3-1】 试比较 filter 滤波与传递函数估计的频率响应。

```
Fs = 1000; % sampling frequency
t = 0:1/Fs:1; % one second worth of samples
xn = sin(2 * pi * 50 * t) + 2 * sin(2 * pi * 120 * t) + randn(size(t));
h = ones(1,10)/10; % moving average filter
yn = filter(h,1,xn);
[HEST,f] = tfe(xn,yn,256,Fs,256,128,'none');
H = freqz(h,1,f,Fs);
```



(a)实际传递函数幅值响应



(b)估计传递函数幅值响应

图 10.4.3-2

```
plot(f,abs(H));
plot(f,abs(HEST));
```

图 10.4.3-2 是两种情况的频率响应图形。

5. cohere(求相干系数函数)

信号 $x(n)$ 和 $y(n)$ 的相干系数函数定义为

$$C_{xy}=\frac{|P_{xy}(\omega)|^2}{P_{xx}(\omega)P_{yy}(\omega)} \tag{10.4.3-3}$$

相干系数反映了随机过程 x(n) 和 y(n) 的统计特性之间的关系。Cxy 的值域是 [0,1]。

语法：

```
Cxy = cohere(x,y,nfft,Fs,window,noverlap)
```

下面语句求【例 10.4.3-1】中 $x(n)$ 和 $y(n)$ 的相干性：

```
cohere(xn,yn,256,Fs,256,128,'none')
```

图 10.4.3-3 显示了 x(n) 和 y(n) 的统计特性在不同频率下的关系。

10.5 窗　函　数

在信号处理中，由于信号本身的特征及计算方法、手段的限制，不可避免地要对数据进行截短。将一个长序列截短成有限长的短序列，必然用到窗函数。对一定的信号选用适当的窗函数，可以减小吉布斯现象的影响。因此，加窗的目的有两个：a. 截短信号序列；b. 减小吉布斯现象的影响。表 10-6 是 MATLAB 信号处理工具箱提供的窗函数。

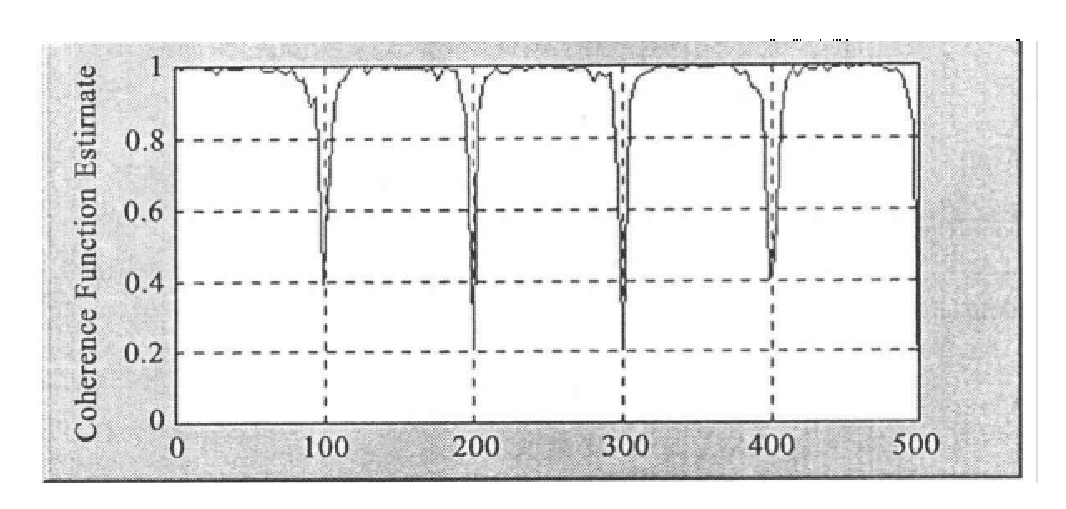


图 10.4.3-3

10.5.1 基　本　窗

矩形窗是基本窗

w=boxcar(n);

产生长度为n的矩形窗，矩形窗的用途仅是截短数据。通常MATLAB以列向量的方式存储窗，因此上述语句等价于：

w=ones(n,1);

表 10-6

窗	函　数	窗	函　数
巴特利特窗	bartlett	海明窗	hamming
布莱克曼窗	blackman	汉宁窗	hanning
矩形窗	boxcar	凯瑟窗	kaiser
契比雪大窗	chebwin	三角窗	triang

巴特利特窗(或三角窗)是两个矩形窗的卷积。函数bartlett和triang得出的结果都是三角窗，但有三点不同：

1)bartlett返回的序列两端是零。因此，对于奇数n，bartlett(n+2)去掉两端的部分和triang(n)是一样的。

```
    bartlett(7)
ans =
0
```

```
0.3333
0.6667
1.0000
0.6667
0.3333
0
    triang(5)
ans =
0.3333
0.6667
1.0000
0.6667
0.3333
```

2)对于偶数,bartlett 仍是两个矩形窗的卷积,而 triang 无标准定义。

3)两者的傅里叶变换不同。对于偶数,bartlett 的傅里叶变换是负的,而 triang 总是非负的。

10.5.2 升余弦窗

矩形窗、海明窗、汉宁窗、布莱克曼窗是升余弦窗的特例,分别是频率为 0,2π/N,4π/N(N 是系列长度)的余弦序列的组合。对于这些窗,若要生成长度是 N 的序列,除了用表 10-6 中的函数外,也可以用下面语句:

```
ind = (0:n-1)' * 2 * pi/n;
w = A-B * cos(ind) + C * cos(2 * ind);
```

系数 A=0.5,B=0.5,C=0 时是汉宁窗。汉宁窗旁瓣小,衰减快,泄漏较矩形窗小,因此纹波较小;A=0.54,B=0.46,C=0 时是海明窗,海明窗的特点是第一旁瓣非常小,泄漏很小,但其他旁瓣没有汉宁窗衰减快;A=0.42,B=0.5,C=0.08时是布莱克曼窗,其主瓣最宽,分辨率稍差,但旁瓣小且衰减快。

10.5.3 凯瑟窗及其应用

1. 凯瑟窗

$$w(n)=\frac{I_0\left[\beta\sqrt{1-\left(1-\frac{2n}{N-1}\right)^2}\right]}{I_0[\beta]}\quad 0\leqslant n\leqslant N-1 \quad (10.5.3\text{-}1)$$

式(10.5.3-1)是凯瑟窗的数学表达式,其中 $I_0[.]$是第一类修正的零阶贝塞尔函数,β 是与 N 有关的参数,选择合适的 β 值,可产生不同的过渡带宽和阻带衰

减。对于长度为 N 的窗，函数 kaiser(n,beta)中的 beta 控制旁瓣的高度。

在 MATLAB 中，用 w= kaiser(n,beta)产生 w 序列。

下面构造 beta 取不同值时的凯瑟窗。

```
n = 50;
w1 = kaiser(n,1);
w2 = kaiser(n,9);
[W1,f] = freqz(w1/sum(w1),1,512,2);
[W2,f] = freqz(w2/sum(w2),1,512,2);
plot(f,20 * log10(abs([W1 W2])))
```

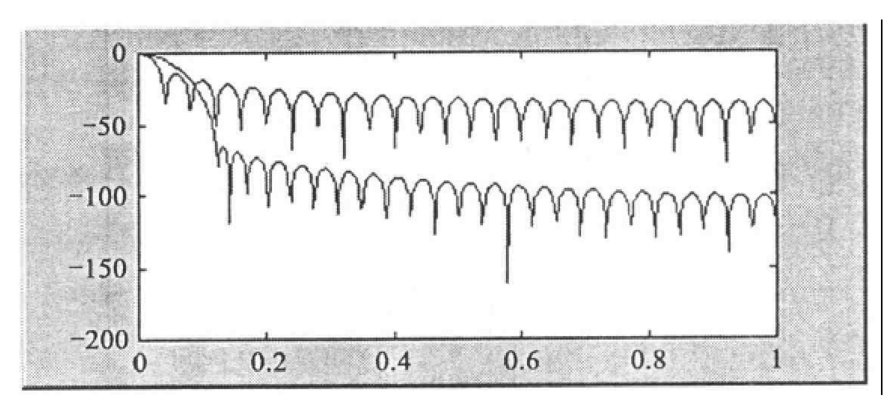


图 10.5.3-1

从图 10.5.3-1 可以看出，beta 的值越大，凯瑟窗的旁瓣越小，但主瓣的宽度也随之加大。对于具体信号，可以根据实际情况，选择 beta，满足选择性和纹波的要求。

2. FIR 设计中的应用

用凯瑟窗设计滤波器，从两种性能指标上满足设计要求：

(1)以旁瓣幅值为设计指标

为得到一 αdB 的旁瓣高度，参数 β 的设定如下：

$$\beta = \begin{cases} 0.1102(\alpha - 8.7) & \alpha > 50 \\ 0.5842(\alpha - 21)^{0.4} + 0.07886(\alpha - 21) & 50 \geqslant \alpha \geqslant 21 \\ 0 & \alpha < 21 \end{cases}$$

(2)以过渡带宽 Δωrad/s 为设计指标

要求过渡带宽是 Δωrad/s，窗口长度

$$n = \frac{\alpha - 8}{2.285\Delta\omega} + 1$$

时，大致可满足要求，但设计后，还需验证。

下面设计一个截止频率为0.5πrad/s，过渡带宽为0.2πrad/s，阻带衰减为40dB的低通滤波器。

```
[n,wn,beta] = kaiserord([0.4 0.6] * pi,[1 0],[0.01 0.01],2 * pi);
b = fir1(n,wn,kaiser(n+1,beta),'noscale');
```

上面的程序段用函数kaiserord计算凯瑟窗的n和β，从滤波器的频率响应曲线(图10.5.3-2)，可看出设计满足要求。

```
[H,f] = freqz(b,1,512,2);
plot(f,20 * log10(abs(H))), grid
```

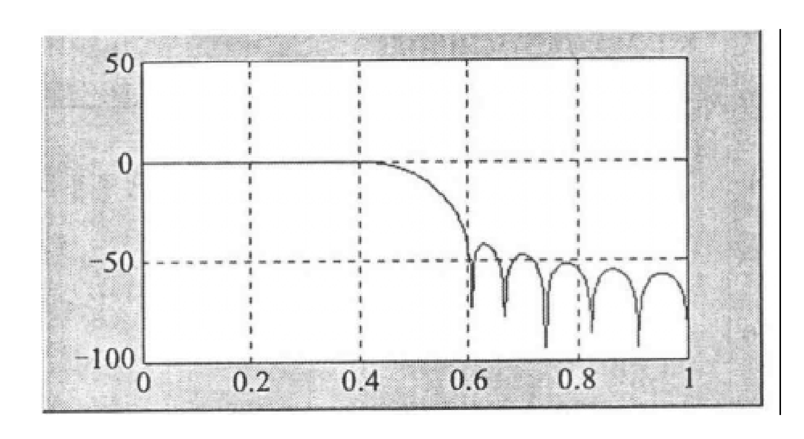


图10.5.3-2

10.5.4 契比雪夫窗

在旁瓣幅值相同的情况下，契比雪夫窗的主瓣最窄，具有等波性，即所有旁瓣幅值相等。

语法：

```
w = chebwin(n,r)
```

参数n-窗长度，r-旁瓣高度。

```
n = 51;
r = 40; % sidelobe height in decibels
w = chebwin(n,Rs);
stem(w)
```

图10.5.4-1是窗的图形，可以看出在输出采样点上有较大的值。下面绘出其频响图形，看在−40dB处的等波性。

```
[W,f] = freqz(w,1,512,2);
plot(f,20 * log10(abs(W)/sum(w))), grid
```

图10.5.4-2是频率响应图形。

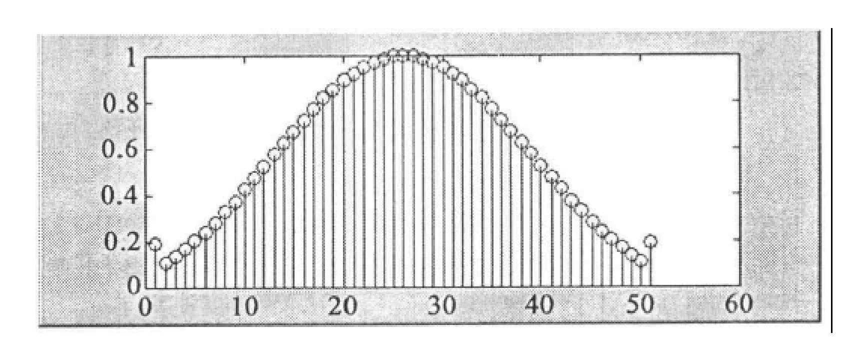


图 10.5.4-1

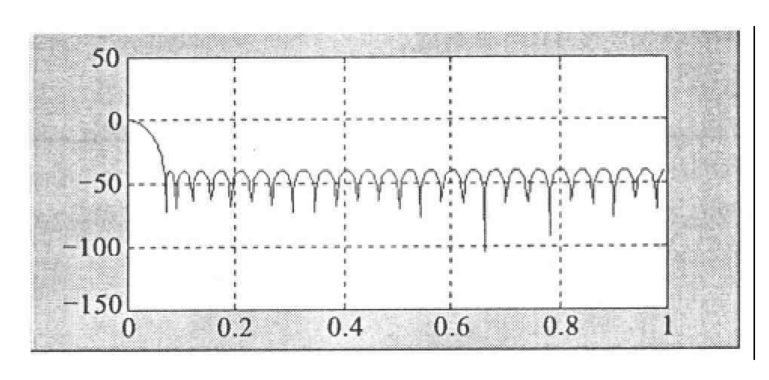


图 10.5.4-2

10.6 系统模型参数估计

参数建模方法利用已知的系统信息，估计系统的数学模型参数，建立系统的数学模型。MATLAB 的参数化建模函数适于有理传递函数，给出未知系统的有关信息，这些函数可以求出系统的线性模型系数。表 10-7 列出了信号处理工具箱中的有关函数。

yulewalk 函数适于 ARMA 模型，在 10.3.1 节已经介绍，不在赘述。

10.6.1 时域模型

lpc，prony，stmcb 求出与给定时域脉冲响应近似的数字有理传递函数的系数。这几个函数的区别在于采用的算法的复杂程度和所得模型精度不同。

1. 线性预报

线性预报算法假设：系统在 k 时刻的采样输出 $x(k)$ 是过去 n 个输出的线性组合（因此称为线性预报），且系数是常数，因此

表10 7

作用域	函数	说明
时域	lpc,levinson prony stmcb	线性预报编码,生成脉冲响应与给定序列匹配的全极点滤波器。 生成脉冲响应与给定序列匹配的IIR滤波器。 给定输入序列,生成输出与给定输出序列匹配的IIR滤波器。
频域	invfreqz, invfreqs	给定复频率响应,生成模拟或数字滤波器的系数。

$x(k)=-a(2)x(k-1)-a(3)x(k-2)-\cdots-a(n)x(k-n-1)$

信号$x(n)$的n阶全极点模型可用lpc估计。

语法:

```
a = lpc(x,n)
```

参数说明:x-信号输入,n-模型阶数

下面说明lpc的用法。首先生成采样信号x,该信号是带附加白噪声的全极点滤波器的脉冲响应。

```
randn('seed',0)
x = impz(1,[1 0.1 0.1 0.1 0.1],10) + randn(10,1)/10;
```

求出4阶系统的系数:

```
a = lpc(x,4)
a =1.0000 0.0395 0.0338 0.0668 0.1264
```

计算中,lpc先调用函数xccor计算x的有偏自相关估计,然后用levinson函数求出系统模型系数a。上面lpc求a的全过程是:

```
r = xcorr(x);
r(1:length(x)-1) = []; % remove corr. at negative lags
a = levinson(r,4)
a =1.0000 0.0395 0.0338 0.0668 0.1264
```

也可以给levinson不同的自相关估计,求出系数a。下面的例子传给levinson的r是x的无偏自相关估计。

```
r = xcorr(x,'unbiased');
```

```
r(1:length(x)-1) = []; % remove corr. at negative lags
a = levinson(r,4)
a =1.0000 0.0554 0.0462 0.0974 0.2115
```

2. Prony 法(ARMA 模型)

Prony 根据给定系统分子、分母的阶次以及脉冲响应构造 IIR 滤波器。

语法:

```
[b,a] = prony(x,nb,na)
```

参数说明:x-给定的信号序列,nb,na-模型阶次。

prony 返回 IIR 滤波器分子和分母的系数,所构造的滤波器的脉冲响应可以充分逼近 x。

下面构造一个 3 阶 IIR 滤波器,x 作为给定序列。

```
[b,a] = prony(x,3,3)
b =1.1165－0.2181－0.6084 0.5369
a =1.0000－0.1619－0.4765 0.4940
```

用 impz 比较滤波器的脉冲响应与原序列 x 的符合程度。

```
format long
[x impz(b,a,10)]
ans =
1.11649535105007      1.11649535105007
－0.03731609173676   －0.03731609173676
－0.08249198453223   －0.08249198453223
－0.04583930972315   －0.04583930972315
－0.14255125351637   －0.02829072973977
0.20400424807471      0.01433198229497
0.02685697779814      0.01148698991026
0.18956307836948      0.02266475846451
0.02717716288172      0.00206242734272
0.08057060786906      0.00545783754743
```

前 4 个数据是完全一样的。若想 prony 构造的滤波器的响应非常近似给定序列,可以用巴特沃思滤波器的脉冲响应作为 prony 的 x。

```
[b,a] = butter(4,2);
h = impz(b,a,26);
[bb,aa] = prony(h,4,4);
```

运行上面的语句,将可以看到 prony 构造的滤波器的脉冲响应与 x 的误差小于 10^{-13}。

3. Steiglitz-McBride 法(ARMA 模型)

函数 stmcb 执行 Steiglitz-McBride 迭代法。Steiglitz-McBride 迭代是对于给定时域脉冲响应,构造 IIR 滤波器的一种算法。

语法:

```
[b,a] = stmcb(x,u,nb,na,niter)
[b,a] = stmcb(x,u,nb,na,niter,ai)
```

参数说明:u-给定系统的输入,与 x 同样长度;niter-迭代因子,缺省值是 5;ai-分母系数的初始估计。缺省方式时,stmcb 和 prony 是一样的。

```
[b,a] = stmcb(x,3,3)
b =
1.1165 -0.6213 -0.8365 1.3331
a =
1.0000 -0.5401 -0.6109 1.1298
```

加参数 u,根据给定的输入、输出序列求系统的系数。

```
y = filter(1,[1 1],x); % Create an output signal
[b,a] = stmcb(y,x,0,1)
b = 1
a = 1 1
```

Steiglitz-Mcbride 法是可以同时求出系统分子、分母系数的快速迭代算法。该算法通常收敛很快,但若模型的阶次很高时,可能不收敛。

如果在建模时遇到困难,可以调整 stmcb 的参数。下面一例的 stmcb 调用,用了参数迭代因子 niter 和模型分母系数的初始估计值 ai。

```
n = 10; % number of iterations
a = lpc(x,3); % initial estimates for denominator
[b,a] = stmcb(x,3,3,n,a);
```

现在比较 lpc,prony,stmcb 的响应与给定信号的误差。

```
a1 = lpc(x,3);
[b2,a2] = prony(x,3,3);
[b3,a3] = stmcb(x,3,3);
[ x-impz(1,a1,10) x-impz(b2,a2,10) x-impz(b3,a3,10) ]
ans =
0.1165         0          0
-0.0058        0          -0.0190
-0.0535        0.0000     0.0818
0.0151         -0.0000    -0.0176
-0.1473        -0.1143    -0.0476
```

```
0.2005    0.1897     0.0869
0.0233    0.0154    -0.0103
0.1901    0.1669    -0.0093
0.0275    0.0251     0.0294
0.0808    0.0751     0.0022
    sum(ans. ^ 2)
ans =
0.1226 0.0834 0.0182
```

从误差平方和可以看出，对于给定的阶次，stmcb 估计的效果最好，prony 次之。

10.6.2 频域模型

invfreqs 和 invfreqz 是 freqs 和 freqz 的逆过程，它们求出与给定复频率响应相匹配的模拟或数字传递函数。

语法：

```
[b,a] = invfreqs(h,w,nb,na,wt,iter,tol,'trace')
[b,a] = invfreqz(h,w,nb,na,wt,iter,tol,'trace')
```

参数说明：h-给定频率点的复频率响应向量；w-频率向量；na，nb-分母、分子阶次；wt-权向量，与 w 同长；iter-迭代因子；tol-迭代容差；'trace'-显示迭代过程。

下面的例子说明从数字滤波器的频率响应求出滤波器的系数。

```
    [b,a] = butter(4,.4) % design Butterworth lowpass
b =
0.0466 0.1863 0.2795 0.1863 0.0466
a =
1.0000-0.7821 0.6800-0.1827 0.0301
    [h,w] = freqz(b,a,64); % compute frequency resp.
    [bb,aa] = invfreqz(h,w,4,4) % model: nb = 4, na = 4
bb =
0.0466 0.1863 0.2795 0.1863 0.0466
aa =
1.0000-0.7821 0.6800-0.1827 0.0301
```

频率向量 w 的单位是 nπ 弧度/采样点数，向量元素可以不是等间隔的。下面是三阶滤波器的例子：

```
    [bb,aa] = invfreqz(h,w,3,3) % find third-order IIR
bb =
0.0464 0.1785 0.2446 0.1276
```

```
aa =
1.0000 -0.9502 0.7382 -0.2006
```

invfreqs 和 invfreqz 设计实系数滤波器。对于正频率 f 处的数据点，±f 的频率响应都可用于这两个函数。

缺省方式时，infreqz 用等误差法确定最优模型。对下式

$$E = \sum_{k=1}^{n} wt(k)\,|h(k)A(\omega(k)) - B(\omega(k))|^2 \tag{10.6.2-1}$$

求极小值，建立线性方程组，解出系数 b 和 a。式中 $A(\omega(k))$ 和 $B(\omega(k))$ 是频率 $\omega(k)$ 处的多项式系数 a，b 的傅氏变换，n 是频率点数（h 和 w 的长度），$wt(k)$ 是权向量，对不同频率点处的误差加权，语句

```
invfreqz(h,w,nb,na,wt)
```

采用了权向量参数。加权后，不能保证 invfreqz 的结果是稳定的。为使实际频响与理想频响间加权误差平方和最小化，invfreqz 用下式构造算法：

$$E = \sum_{k=1}^{n} wt(k)\,\left|h(k) - \frac{B(\omega(k))}{A(\omega(k))}\right|^2 \tag{10.6.2-2}$$

为使用上式构造的算法，可加入迭代因子参数，说明迭代次数，如

```
wt = ones(size(w)); % create unity weighting vector
[bbb,aaa] = invfreqz(h,w,3,3,wt,30) % 30 iterations
bbb =
0.0464 0.1829 0.2572 0.1549
aaa =
-0.8664 0.6630 -0.1614
```

习　题　十

1. 设计巴特沃思低通数字滤波器。通带带宽 0.3π，阻带边缘频率 0.6π，阻带衰减大于 35dB，采样间隔 0.01 秒。

2. 设计满足给定指标的 IIR 数字滤波器。

1）高通：$f_p=400$Hz，$f_s=300$Hz，$F_s=1000$Hz，$a_p=3$dB，$a_s=35$dB；

2）带通：$f_{sl}=200$Hz，$f_1=300$Hz，$f_2=400$Hz，$f_{sh}=500$Hz，$F_s=2000$Hz，$a_p=3$dB，$a_s=40$dB。

3. 设计一个 70Hz 的 IIR 滤波器，下通带 0～60Hz，阻带在 63Hz，上通带对称。要求通带 2dB，阻带 55dB。

4. 理想低通 FIR 滤波器的频率特性

$$Hd(e^{j\omega}) = \begin{cases} 1 & |\omega| \leqslant \pi/4 \\ 0 & \pi/4 < |\omega| < \pi \end{cases}$$

分别用矩形窗、海明窗设计滤波器，要求具有线性相位，长度 29 点。

5. 设计一长度为 55 的多带 FIR 滤波器。通带频率(归一化)：0.1～0.15，0.3～0.6，其余阻带，阻带边缘频率 0.05，0.18，0.25，0.41，并自给通带、阻带的加权值，研究不同加权值对滤波器性能的影响。

6. 用 Welch 法估计 512 点的白噪声序列的功率谱(用不同的段，不同的重叠量)，并与周期图法的功率谱曲线比较。

7. 用三阶 FIR 系统

$$H(z) = 1 - 0.1z^{-1} + 0.09z^{-2} + 0.64z^{-3}$$

的脉冲响应加白噪声，产生 20 点序列 x，分别用 lpc，prony，stmcb 估计系统参数，模型阶次 $n=5$，并比较三者估计精度。

第十一章　动态仿真系统 SIMULINK

11.1 引　　言

SIMULINK 是 MATLAB 实现动态系统建模、仿真的一个集成环境，它使 MATLAB 的功能得到进一步的扩展。这种扩展的意义表现在：(1)实现了可视化建模。在 SIMULINK 窗口，用户通过简单的鼠标操作就可建立起直观的系统模型，并进行仿真；(2)实现了多工作环境间文件互用和数据交换，如 SIMULINK 与 MATLAB，SIMULINK 与 C、FORTRAN，SIMULINK 与 DSP，SIMULINK 与实时硬件工作环境等的信息交换都可以方便地实现；(3)把理论研究和工程实现有机地结合在一起。

本章先通过一个简单的例子介绍模型创建、仿真的大致过程，然后详细介绍 SIMULINK 的主要操作。仿真算法的选择、参数设置以及实施仿真的不同操作方法是 11.3 节的内容。11.4 节介绍仿真系统的线性化模型；如何从非线性模型获取平衡工作点和线性化模型；如何从连续-离散混合模型获得任意采样频率下的等效模型。活用 SIMULINK 的关键在 S 函数，这部分内容比前面几节更深入 SIMULINK 的核心。读者有了前面的基础，就较容易掌握 S 函数。关于 S 函数的内容被安排在最后一节。

11.1.1 SIMULINK 的安装

SIMULINK 的安装与 MATLAB 的安装类似。在 MS-Windows 下的安装方法，可以参考第二章关于 MATLAB 的安装。SIMULINK 安装完成后，安装程序自动在 MATLAB 的目录下建立 SIMULINK 目录，并修改 MATLAB 的启动文件 matlabrc.m。

11.1.2 SIMULINK 入门

下面以一简单系统为例，介绍包括建模和仿真在内的 SIMULINK 的基本操作：

1. 在 MATLAB 窗口输入命令 simulink，打开 SIMULINK 模块库浏览器(图 11.1.2-1)。



图 11.1.2-1　SIMULINK 模块库浏览器

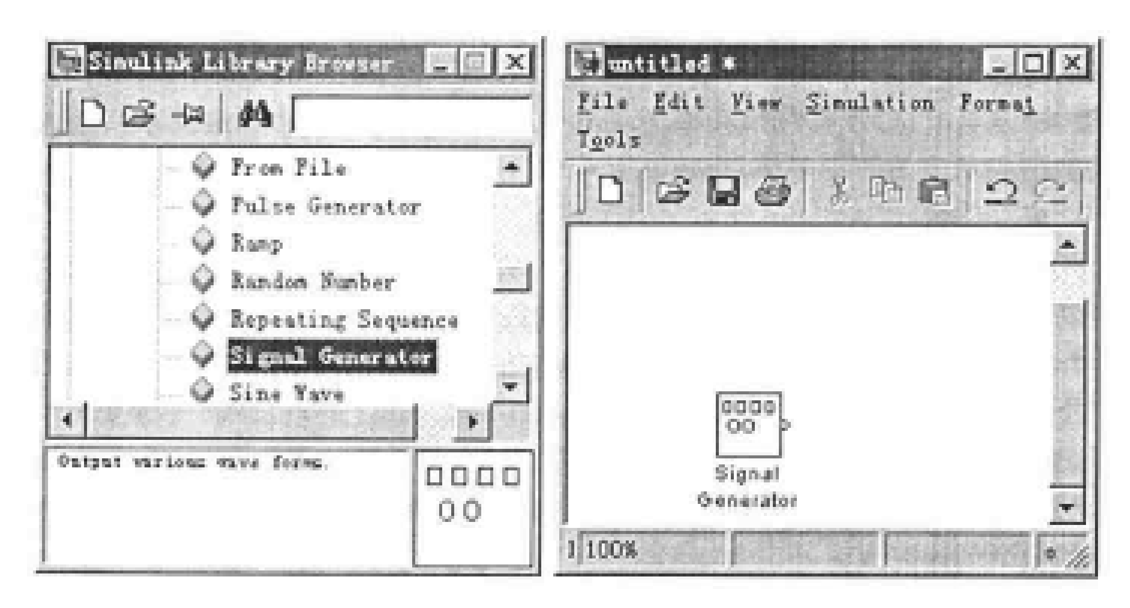


图 11.1.2-2　复制信号发生器模块

2. 选择工具栏上 Create a new model 按钮，建立名为"Untitled"的模型窗口。

3. 用鼠标双击 Simulink 图标，再双击信号源 Source 图标打开信号源模块库。

4. 移动鼠标的光标到信号发生器图标，按住鼠标左键，将它拖到"Untitled"窗口。信号发生器(Signal Generator)模块从子库窗口复制到"Untitled"窗口

(图 11.1.2-2)。

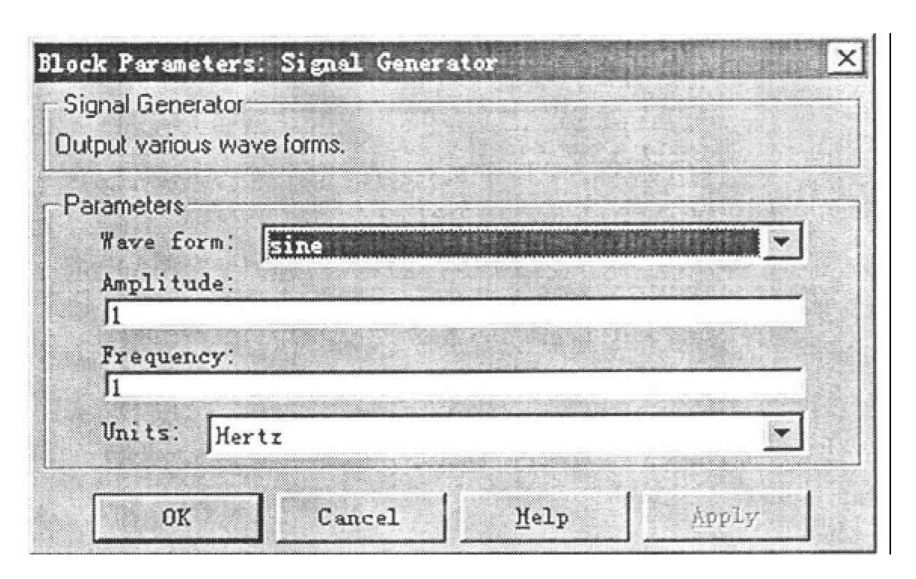


图 11.1.2-3 信号发生器参数设置对话框

5. 设置模块内部参数。被复制到新窗口中的模块具有缺省的参数设置。如果想改变信号发生器的参数设置,用鼠标双击它的图标,将弹出一个如图 11.1.2-3 所示的对话框,显示信号波形、幅值和频率的设置。在图 11.1.2-3 中,加亮的 sine (正弦波)是当前选取的波形,频率(Frequency)是 1,幅值(Amplitude)为 1,输入框中的数值表示当前的设置值,可以在其中输入数值重新设置,完成参数设置后,单击 OK 保存设置。

用同样的方法打开输出模块库(Sinks),从中选取示波器(Scope),然后把它复制到新的系统窗口中,如图 11.1.2-4 所示。Signal Generator 和 Scope 图标中的符号“>”表示信号的输入和输出端,指向模块内的是输入端,指向模块外的则是输出端。下面连接这两个模块,移动鼠标到 Signal Generator 的输出端,光标变为“+”号后,按住左键拖动十字图符到 Scope 的输入端,然后释放左键,带箭头的连线将两模块的输入和输出端连上,箭头表示信号的流向(见图 11.1.2-5)。至此,一个最简单的模型创建完成。

6. 仿真操作一:信号发生器的参数设置。双击示波器图标后,显示示波器(图 11.1.2-6)。在 Simulation 菜单上,单击 Start 命令启动仿真过程。在虚拟示波器上将看到信号发生器输出的波形。如果看不到希望的波形,可调整示波器的时间范围和幅值范围。

7. 仿真操作二:仿真参数的设置。在 SIMULINK 中,仿真过程中动态数据的计算都是由数值积分实现的。尽管本例从信号源到示波器没有其他环节,但动态数

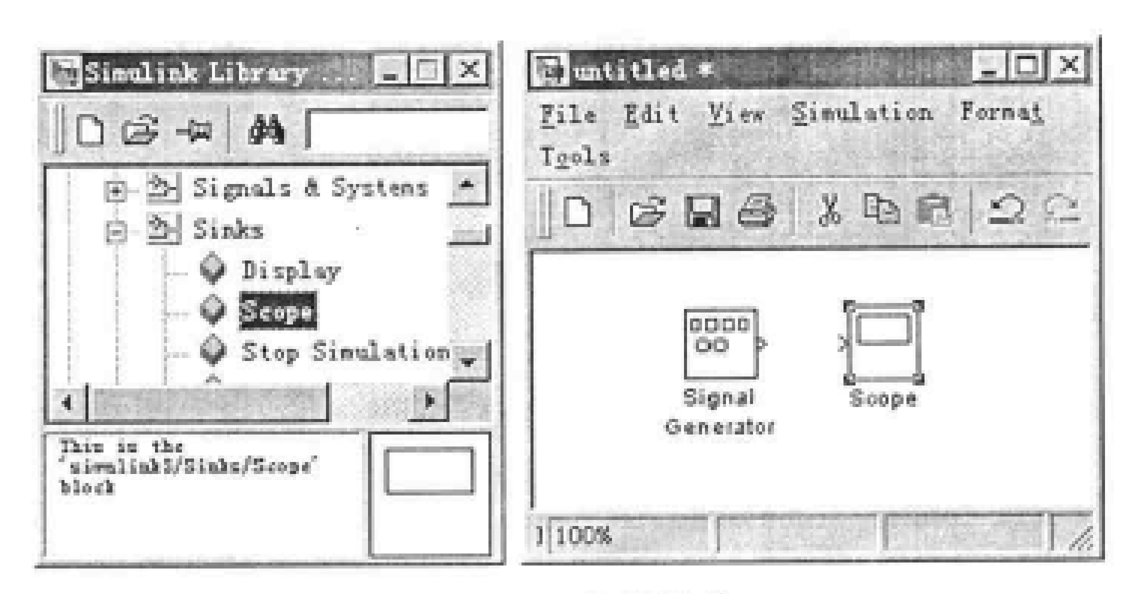


图 11.1.2-4　复制示波器

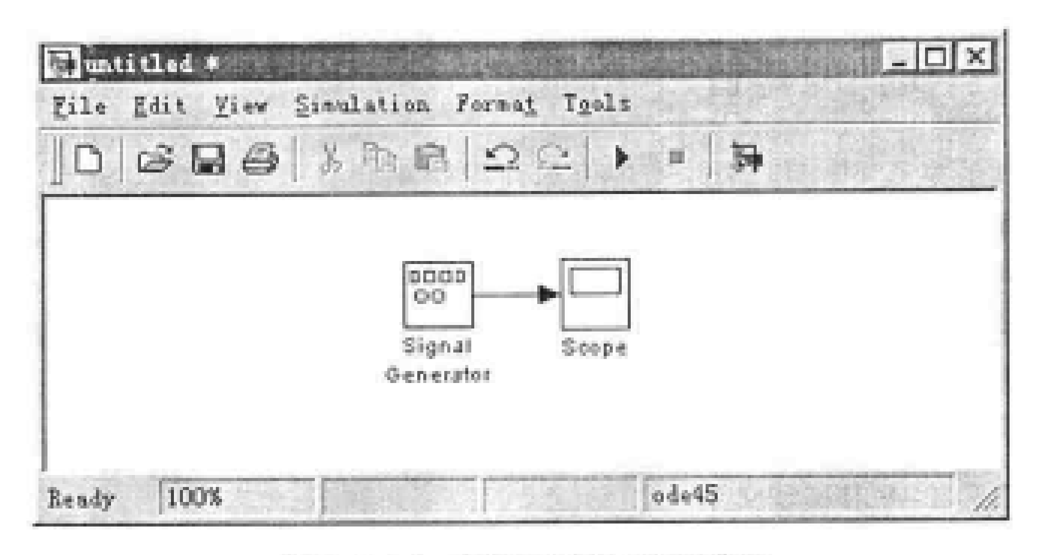


图 11.1.2-5　连接信号发生器和示波器

据仍是经数值积分计算而得的。因此，要得到满意的输出效果，应当选取适当的仿真参数。在仿真窗口中选 Simulation 菜单，单击 Parameters，窗口弹出 Simulation Parameters 对话框，可以看到对话框有不少选项，改变这些选项可以得到不同的仿真结果。例如 Solver 页中的 Max step size 属性表示最大积分步长，使用不同步长仿真，结果是不一样的。步长选择不当，仿真结果可能不理想，一般使用缺省设置，最大步长为 auto 即可获得满意的输出。过小的步长将消耗大量内存，增加仿真

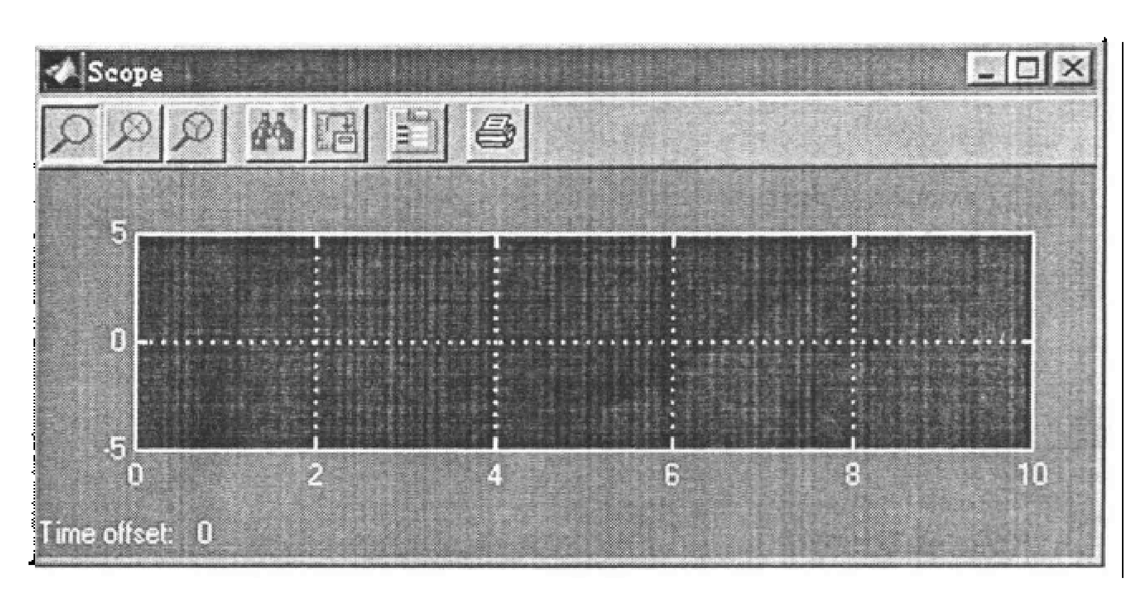


图 11.1.2-6 示波器的控制面板

时间。

11.2 模型的构造

为了掌握 SIMULINK，应学习如何操作模块以及如何建立模型。对于 SIMULINK 的模块库(图 11.1.2-1)的内容不作介绍，本节着重介绍构造模型的方法及过程。SIMULINK 完全采用方框图的“抓取”功能来构造动态系统。系统的创建过程就是绘制方框图的过程。在 SIMULINK 环境中方框图的绘制完全依赖于鼠标操作。鼠标的不同操作体现在光标的形状上。在缺省状态下鼠标是一个箭头，但在图形的操作中鼠标呈现不同的形状。

11.2.1 创建模型文件

建立 SIMULINK 模型通常有三种方式：

1. 直接从 MATLAB 窗口选取 File，单击 New，选取 Model，MATLAB 会打开一个新的名为“Untitled”的模型编辑窗。当然也可像第 11.1.2 节中介绍的那样，在 MATLAB 窗下输入 simulink 命令，打开 SIMULINK 模块流览器，选择工具栏上 Create a new model 按钮，建立“Untitled”模型窗口。

2. 如果仿真模型已经存在，那么在 MATLAB 窗口下直接键入模型文件名称，便可打开该模型，用户可以对它进行编辑、修改和仿真。

3. SIMULINK 模型是扩展名为 mdl 的 ASCII 码文件，因此可以使用字处理

软件对其创建、修改和编辑，这同建立一个普通 M 文件的过程一样，但要符合模型文件的格式。

11.2.2 标准模块的选取

由图 11.1.2-1 可见，SIMULINK 模块库按模块类型分为若干个子库。用鼠标双击子库便可见到库中存放的各种标准模块。这些模块都可被复制到用户的模型窗中(见第 11.1.2 节中的第四步)。

11.2.3 模块的移动、删除和复制

1. 移动模块

将光标置于待移动模块上，按住鼠标左键将模块拖到合适的位置即可。模块移动时，它与其他模块的连线也随之移动。

2. 选定模块

模块选定操作是许多其他操作(如删除、剪切、复制)的"前导性"操作。选定模块的方法有两种：

(1) 用鼠标单击待选模块，模块四个角处出现黑块控点，表示已经选定。

(2) 如果选择一组模块，可以按住鼠标按钮拉出一个矩形虚线框，将所有待选模块包在其中，然后松开按钮，则矩形里所有的模块同时被选中。

3. 删除、剪切和复制模块

对选定模块的删除、剪切和复制操作如下：

(1) 删除：按 Delete 键，删除选定的模块。

(2) 剪切：选择 Edit 菜单中 Cut 命令，将选定模块移到 Windows 的剪贴板上，可用 Paste 命令粘贴。

(3) 复制：在 Edit 菜单中单击 Copy，然后将光标移到欲粘贴的地方，单击鼠标左键，再选 Edit 中 Paste，就会在选定的位置上复制出相应的模块。新复制的模块和原模块的名称会自动编号，以资区别。另外按住 Ctrl 键，用鼠标把待复制模块拖到希望位置后，松开鼠标左键，也可以复制模块。

11.2.4 连接模块

关于模块连接，在本章一开始就已经涉及。要注意模块的输入、输出口和各模块间的信号流向。在 SIMULINK 中，模块由输入口接受信号，从输出口发送信号。

【例 11.2.4-1】 建立如图 11.2.4-1 所示的模型。

(1) 单击 SIMULINK 模型窗中 File，选 New 开一个新的"Untitled"窗口。

(2) 分别从信号源库(Source)、连续系统模块库(Continuous)、数学库

(Math)、输出模块库(Sinks)中,把信号发生器(Signal Generator)、传递函数模块(Transfer Fcn)、求和模块(Sum)和示波器(Scope)拖到“Untitled”窗口。各模块的位置如图11.2.4-1所示。

(3)图11.2.4-1中模块间的连线有两类:一类是“直”线,它从某模块的输出口出发直指另一模块的输入口。这种连线的生成方法在第11.1.2节中已经讲过。另一类连线是“折”线。这类“折”线的生成方法是:把鼠标光标移到传递函数模块和示波器间连线的中点附近,按下鼠标右键,光标由箭头变为十字,往下拉动鼠标到适当位置后放开右键,屏幕上就出现一条由中点引出的箭头线,再从此箭头开始用鼠标水平向左画线到适当位置,再松开鼠标键,照此操作,直到整个“折”线绘完。

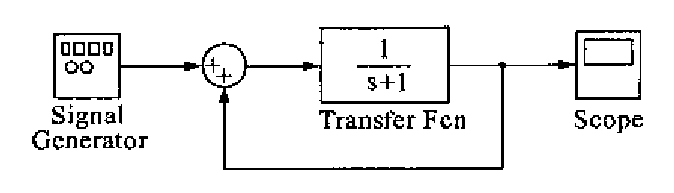


图11.2.4-1 简单模型

11.2.5 改变模块属性

模块的属性分为两种:模块的标题和模块的内部参数。为适合仿真需要,模型中所复制的标准模块的标题和参数常需修改,下面就介绍修改方法。

1. 修改标题

模块标题是指标识模块图标的字符串。对用户所建模型窗中模块标题进行修改的具体方法如下:

(1)用鼠标单击标题,标题即变成输入框。

(2)输入新的标题。

(3)鼠标移出输入框,单击编辑窗口中的任一地方,修改工作结束。

2. 模块内部参数的修改

双击待修改参数模块的图标,打开参数设置对话框,然后改变对话框相应栏目中的数据便可。

【例11.2.5-1】 修改图11.2.4-1的模型。

(1)修改标题 单击传递函数模块标题“Transfer Fcn”,输入汉字字符“传递函数”。再修改示波器模块标题“Scope”,输入汉字字符“示波器”,在窗口空白处单击鼠标左键。

(2)修改求和模块,输入极性,用鼠标双击求和模块(Sum),弹出参数设置对

话框。

（3）把 List of signs 栏中的缺省极性改成如图 11.2.5-1 所示形式，单击 OK，原求和模块图标便成为图 11.2.5-2 所示形式。图 11.2.5-2 为修改后的模型。

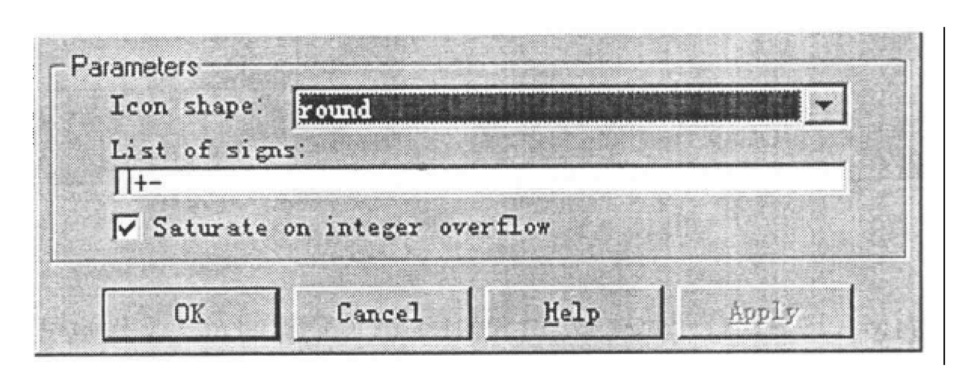


图 11.2.5-1　求和模块对话框

3. 对传递函数模块参数的修改

双击传递函数模块，弹出如图 11.2.5-3 所示对话框。Denominator 栏是传递函数的分母多项式系数向量。把该栏中的缺省值改成图 11.2.5-3 所示的行向量，单击 OK，传递函数模块图标中的函数表达式变为图 11.2.5-2 中的形式。值得指出的是：

（1）在参数设置时，任何 MATLAB 工作内存中已有的变量、合法表达式、MATLAB 语句等都可以填入编辑框。

（2）模块图标的大小是可以用鼠标操作调整的。因此，假如传递函数表达太长，原方框容纳不下，可以用鼠标把它拉到适当大小，使整个方框图标美观易读。

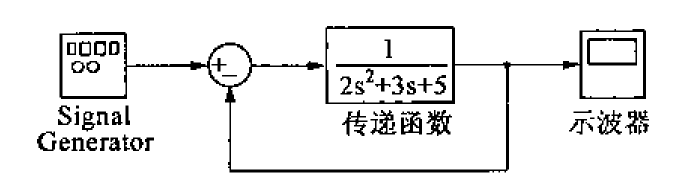


图 11.2.5-2　修改后的模型

11.2.6　保存模型文件

构造好一个模型后，用 File 中 Save 保存模型文件。文件是以 ASCII 码形式存储的 mdl 文件，其中包含了该模型的所有信息（数学模型内涵和外部框图表现）。如【例 11.2.5-1】的全部步骤执行完后，以“simu”名保存，以后在 MATLAB 窗口输

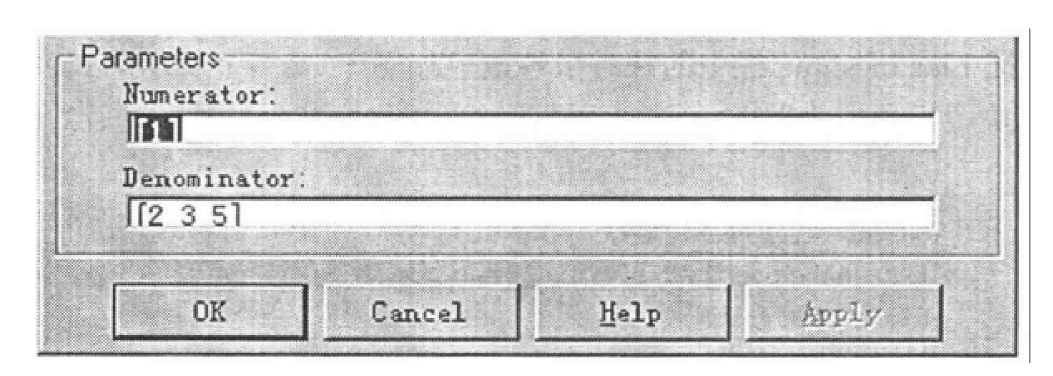


图 11.2.5-3 传递函数模块对话框

入 simu 时，MATLAB 打开此模型。由 SIMULINK 创建的 mdl 文件也可以用 type 命令查看其内容，也可以用字处理器进行修改和编辑。

11.3 数 值 分 析

仿真涉及常微分方程的数值算法。由于动态系统行为的多样性，目前还没有一种算法能够保证所有模型的数值仿真结果总是准确、可靠的。因此，SIMULINK 有一组不同的数值算法供用户选择。为得到正确仿真结果，用户必须针对不同模型，仔细选择算法及仿真参数。

11.3.1 菜单操作方式下仿真算法和参数的选择

所谓菜单操作方式的仿真是指，仿真在 SIMULINK 的模型窗口进行，这种仿真方式最直观。在这种仿真方式下，无论是对模型本身还是对数值算法及参数的选择都可以方便地调整、修改。

1. 模型参数和框图的实时操作

模型不仅在仿真前允许编辑和修改，而且在仿真过程中也允许作一定程度的修改。在菜单操作方式下，允许对模型和框图进行如下实时操作：

（1）仿真模块的参数允许实时修改，限制条件是该参数的变化不改变模型的结构（包括模型几何结构、输入输出维数及状态变量数）。

（2）离散模块的采样时间允许实时修改。

（3）允许用浮动示波器（Floating Scope）实时观察任何一点或几点的动态波形（所谓浮动示波器是指在模型视窗里与系统模型没有任何可见连接线的示波器）。

（4）在一个模型仿真的同时，允许打开另一个进行仿真。

2. 选择算法和设置仿真参数

在仿真模型的SIMULINK窗口设置仿真算法和参数。单击Simulation菜单中Parameters选项，窗口弹出仿真参数对话框(图11.3.1-1)。对话框分为四页，下面介绍Solver页。

Solver页有设置仿真时间、选择仿真算法和算法参数、改变输出方式等选项。

(1) 仿真时间　可以修改Start Time和Stop Time的值，改变仿真时间。注意仿真时间不是机器运行仿真模型的时间，而是实际系统运行的时间。

(2) 仿真算法　SIMULINK提供若干种数值积分算法，其中变步长(Variable-step)算法8种，固定步长(Fixed-step)算法6种，仿真时可以选择不同算法。一般情况下，系统设置的缺省参数已能较好地解决问题，但有时改变缺省算法，有助于提高仿真精度和效率。

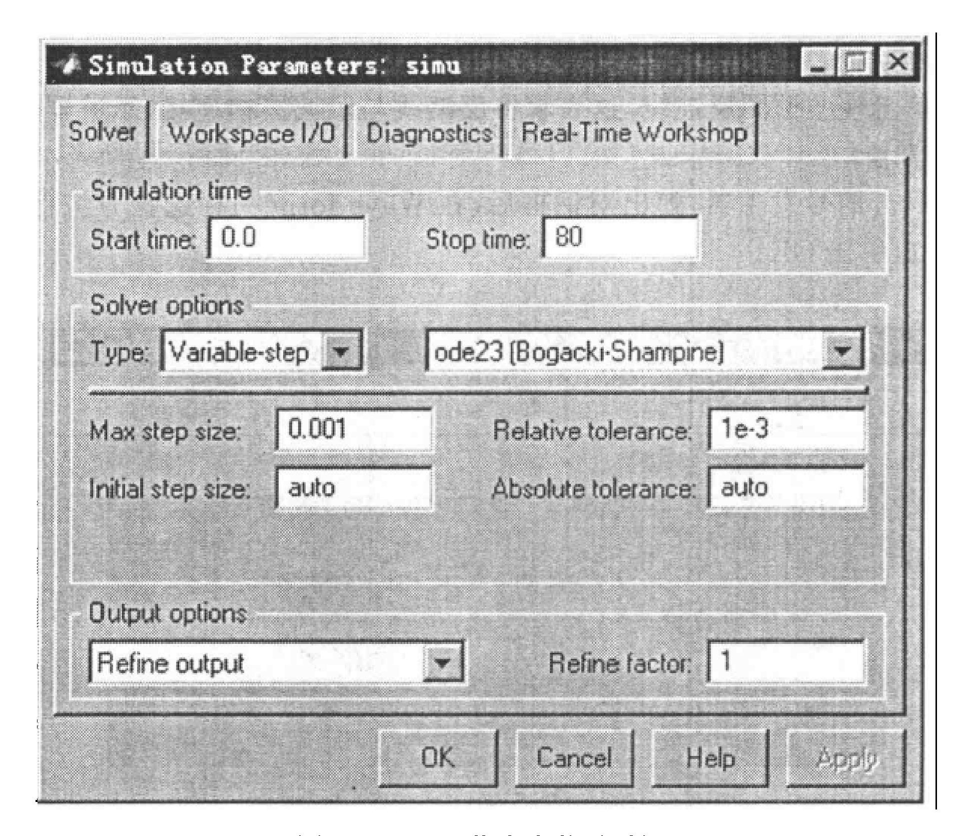


图11.3.1-1　仿真参数对话框

(3) 算法步长　SIMULINK根据

$h=(\text{stoptime}-\text{starttime})/50$

设置最大步长(Max step size)。如果认为所选算法可能遗漏重要仿真结果，可以减小最大步长。所选算法根据模型初始状态选择初始步长(Initial step size)，若认为

不当，也可修改。

(4) 容差(Relative tolerance Absolute tolerance)控制计算误差。积分算法采用局部误差控制技术检测仿真过程每一时间段的误差。如果误差过大，将自动修改步长，重新计算，使误差满足设定值要求。

(5) 输出选择　如果输出图形太粗糙，可以改变 Refine output 的值，增加输出点数，使曲线更平滑。Produce additional output 可以指定附加的时间点输出，此选项改变仿真步长，以使其与指定的附加时间点一致。Produce additional output only 改变原算法的输出，仅在指定的时间点输出。在比较不同的仿真结果，要求各仿真都有相同的时间点输出时，此选项很有用。

【例 11.3.1-1】　使用浮动示波器。

(1) 双击示波器，单击示波器面板上 Properties 按钮，在示波器特性对话框的 floating scope 选项上打勾，单击 OK，与示波器输入端相连的箭头断开，示波器处于浮动工作状态。

(2) 单击模型框图反馈线，这表示将反馈线与示波器相连接，示波器输出的是反馈线的信号。

(3) 双击信号发生器，弹出对话框后，在 Wave form 一栏选择方波信号，设置频率为 0.1，单击 OK。

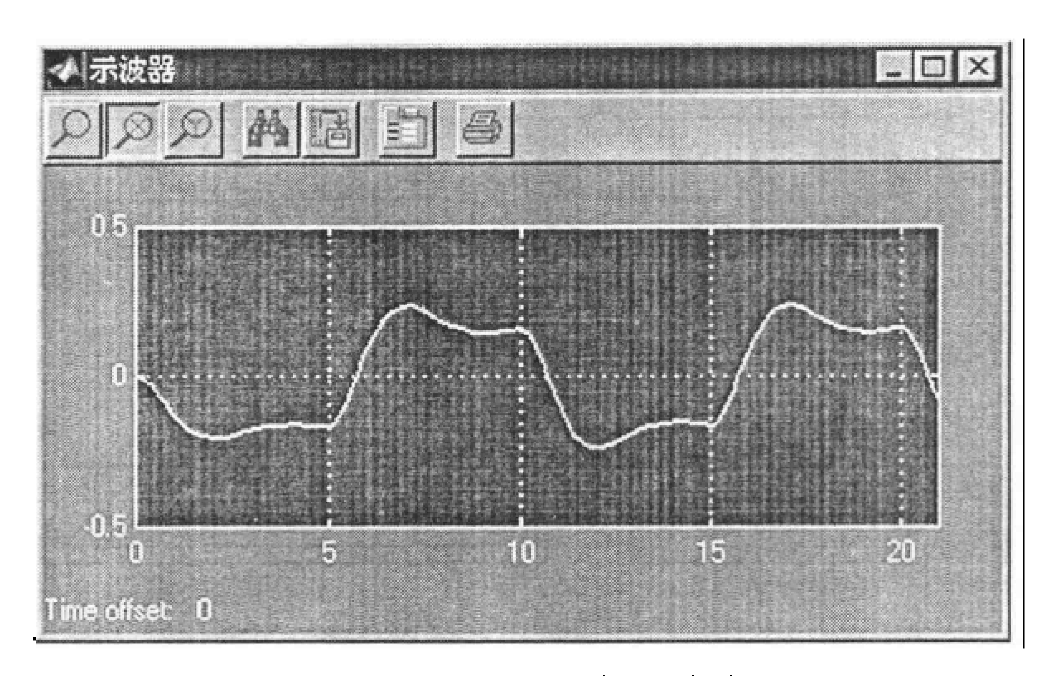


图 11.3.1-2　用浮动示波器观察波形

(4) 选择 Simulink 菜单中的 Start 开始仿真，在示波器中就可以看到图 11.3.1-2所示的波形。读者可以试着实时地调整模型参数、算法及算法参数，观察

波形的变化。若要停止仿真，单击 Simulink 菜单中的 Stop 命令。

11.3.2　仿真的命令操作方式

1. 在 MATLAB 窗口进行仿真

在 SIMULINK 模型窗口中建立的仿真模型可以在 MATLAB 窗口中被调用、仿真，其调用、仿真由 sim，set_param 命令完成。由于有了这些命令，使我们得以在编制的程序(M 文件)中进行仿真，给编制大型的应用程序带来很大方便。

调用格式：

[t,x,y] = sim(model,timespan,options,ut)

说明：

(1) 形参中仅 model 是必须的，其余参数都可以缺省，缺省参数采用 Simulation Parameters 对话框设置的参数。

(2) t 为仿真时间向量。

(3) x 为仿真状态空间矩阵。

(4) y 为仿真输出矩阵，每列表示一个输出模块的输出，按输出模块号排序。

(5) model 是模型文件名，为字符串变量。

(6) timespan 是仿真时间。Timespan=[tStart tFinal]，一般 tStart=0，tFinal 是仿真结束时间。用 timespan=[tStart tAddtional tFinal]可得到指定时间点的仿真结果，tAddtional 是含有指定时间点的向量。

(7) options 为设置仿真参数，这是用 simset 命令设置的结构。操作方法参见 simset 命令。

(8) ut 为可选的外部输入，给输入模块输入数据。ut 可以是函数(用字符串表示，该函数以 u=UT(t)的形式指定仿真中每个时间的输入)，或是一个对应各输入端口的时间-输入值的表格，表格形式可以是矩阵或结构。

set_param('sys','SimulationCommand','cmd')

sys 是模型文件名，为字符串变量。SimulationCommand 指出后面跟着的是仿真命令，cmd 可以是 start，stop，pause，continue 或 update，注意都是字符串变量。

2. 命令操作仿真前模型输入模块的选择

仿真的命令方式中，阶跃函数信号(Step Input)是最常用的一种信号源，它较适于系统动态特性的研究。

阶跃输入模块的阶跃时刻、“跃前”函数值、“跃后”函数值都可以设置。注意：缺省的阶跃时刻是 1，而不是 0。在命令操作方式的仿真中，常用到 SIMULINK 模块库中的两个模块：输出接口(Output)；数据暂存模块(To Workspace)。下面分别举例说明它们的使用方法。

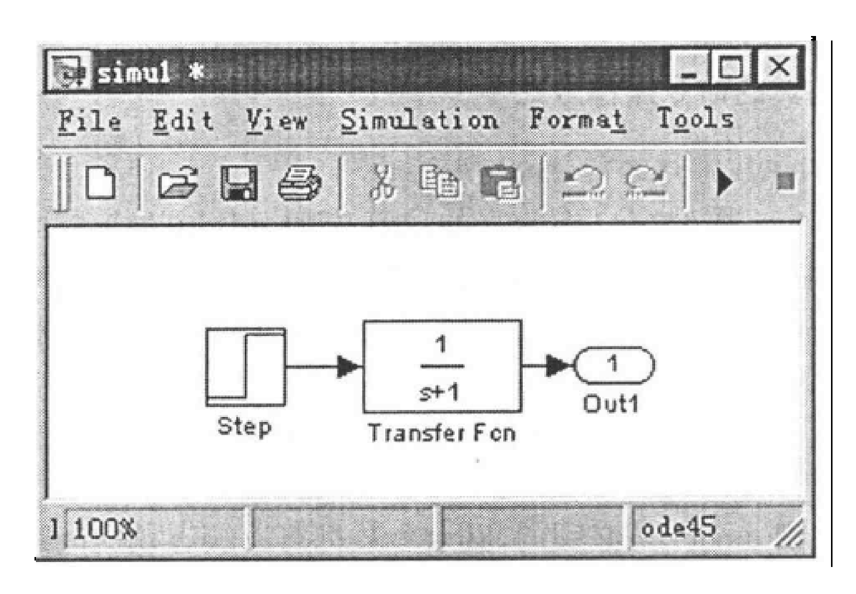


图 11.3.2-1 带输出接口的模型

【例 11.3.2-1】 对图 11.3.2-1 所示模型进行仿真的命令操作方式。

(1) 从信号源库选择阶跃输入模块，信号与系统库选择输出模块复制到模型窗口。

(2) 在图 11.3.2-1 所示的模型视窗里，选择 File 菜单中的 Save，以 simu1 名存盘。

(3) 在 MATLAB 窗口中，运行以下命令就可得到如图 11.3.2-2 所示的输出图形。

```
[t,x,y]=sim('simu1',20);
    plot(t,y)
```

【例 11.3.2-2】 利用数据暂存模块，进行命令操作方式的仿真。

(1) 构造图 11.3.2-3 所示的模型。从输出模块库 Sink 中复制数据暂存模块(To Workspace)。

(2) 双击接在输出端的数据暂存模块，在对话框 Virable Name 栏中填写 y1，将 Save format 栏中的数据格式改为 Matrix，单击 OK。此时模块图标名称已变为 y1。用同样方法把数据暂存模块 1 命名为 t，并改数据格式为 Matrix。模型中要有时钟信号模块，从信号源库复制。

(3) 把所做模型保存为文件 simu2.mdl。

(4) 在 MATLAB 窗口运行以下命令也可得到图 11.3.2-3 所示的响应曲线。

```
sim('simu2',20)
polt(t,y1)
```

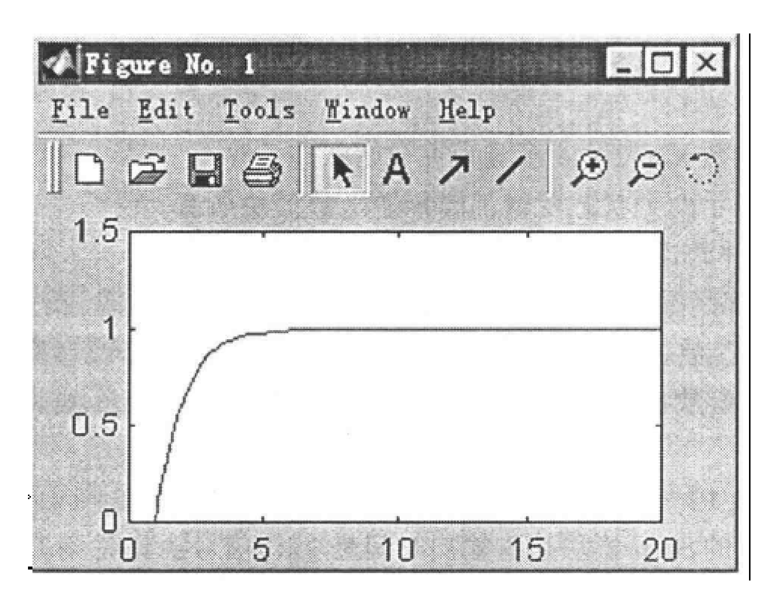


图 11.3.2-2　系统的输出响应曲线

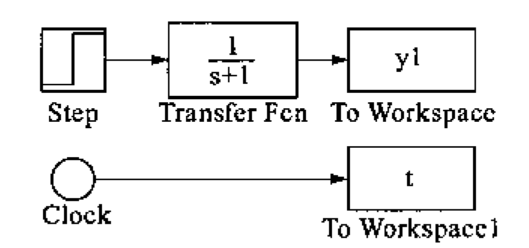


图 11.3.2-3　输出到数据暂存模块的模型

11.3.3　仿真中的几个重要问题

1. 模型变量和初始状态的查询

模型中设置的各种参数:初始条件、状态个数以及输入输出的个数等都很容易查询而得。在此不作一般讲述,而仅通过算例作出说明。

【例 11.3.3-1】　图 11.3.2-1 所示系统的信息查询。

```
[sizes,x0,xstr]= simul                % simul 是文件名
sizes = [1 0 1 0 0 0 1]
x0 = [0 0]
xstr =
/sim/传递函数
    /sim/传递函数
```

说明：

(1) 上述返回参数 sizes(1)= 1 表示连续状态变量数为 1，sizes(3)= 1 表示输出口数为 1。

(2) 返回参数 x0 表示该系统为零初始状态。

(3) xstr 表示两个状态变量都属传递函数模块所有。

2. 初始条件的设置

一般情况都是采用缺省参数进行仿真，但有时为了效率、精度以及习惯等原因，要修改仿真参数。前面已经介绍了在窗口仿真过程中修改参数的方法，在命令行方式下，仿真参数也可以设置，所用命令主要有 simset，simget，set_param，get_param 等。

【例 11.3.3-1】 对图 11.3.2-1 所示系统进行初始参数不同设置的仿真。

MATLAB 的 Step 模块缺省输出时间是 1，即信号源到时刻 1 才有输出。从图 11.3.2-2 可以看到，仿真结果在 0～1 间的值是 0。若将阶跃发生的时刻设在 0，可以改变输出的图形。运行以下命令，可得图 11.3.3-1 所示的相轨迹图。

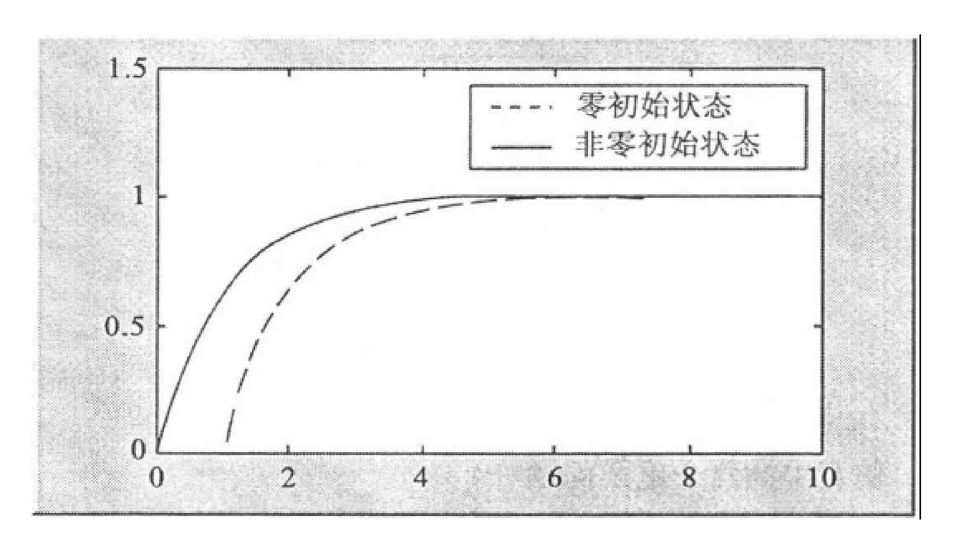


图 11.3.3-1 不同初始条件对相轨迹的影响

```
[t1,x1,y1]=sim('simu1');
set_param('simu1/Step','time','0');
[t2,x2,y2]=sim('simu1');
plot(t1,y1,'r:',t2,y2,'b-');
legend('零初始状态','非零初始状态')
```

说明：第一条语句采用系统默认的初始条件；第二句设置阶跃初始时间在 0 时刻。它们的作用可以从仿真输出上清楚地看到。

3. 代数循环

在构造系统时应注意避免代数循环的发生。所谓代数循环就是几个直通模块构成的一个反馈环节。当发生这种情况后，仿真速度将大大变慢甚至无法进行下去。下面是几个常见的直通模块：

(1) 增益模块(Gain)；

(2) 大多数非线性环节，如查表模块(Look Up Table)、限幅模块(Limiter)等；

(3) 分子分母同阶的传递函数；

(4) D矩阵不为零的状态方程(State-Space)模块。

4. 数据的插补

SIMULINK算法的步长设置为Variable-step时，其步长是自适应的，在计算过程中步长按精度的要求变化。在轨迹平坦的地方步长取得较大，以提高效率，而变化剧烈的地方，步长则取得较小以满足计算精度。因此，对同一个问题，算法不同或同一算法参数不同，将产生出点数和取值位置都不同的计算数据，这就给不同仿真结果的比较带来困难。此外，当计算数据较少，输出曲线不太光滑时，也要用到数据插补技术。在MATLAB中可以用数据插补函数(如interp2，spline)进行插补。

【例11.3.3-2】 利用【例11.3.3-1】修改过的图11.3.3-1模型，进行算法精度控制的比较。

```
[t,x,y]=sim('simu1',10,simset('RelTol',1e-8,'InitialStep',1e-6,'MaxStep',1]);
[t1,x1,y1]=sim('simu1',10,simset('RelTol',0.1,'InitialStep',1e-5,'MaxStep',1]);
vint=table1([t,y],t1);
vector_error=norm([yint-y1])
per_point_error=sum(abs(yint-y1)/length(t1))
vector_error = 0.0133
nper_point_error = 0.0011
```

说明：

(1) 被仿真系统的文件是simu1.mdl，积分时间10秒，初始条件采用默认值。前两条命令分别定义了不同的算法参数：积分相对误差、最小步长和最大步长。

(2) 第三句使用MATLAB的一维线性插补函数table1，由高精度计算结果按低精度数据取值点位置进行插补，从而在相同的数据点上进行两组数据的比较。

5. MATLAB数值仿真算法

对于不同的模型，SIMULINK提供了表11-1中的七种求解微分方程的算法以供选择。

6. 步长控制

在下面仿真命令中：

```
[t,x,y]= sim('simu1',10,simset('RelTol',0.1,'InitialStep',1e-5,'MaxStep',1));
```

用 simset 定义了积分的相对误差、初始步长和最大步长。如何设置这三个参数,对仿真结果的精度有较大的影响。下面是设置这三个参数时,需注意的几个问题。

表 11-1　　SIMULINK 仿真算法

仿真算法	算　法　说　明
ode45	龙格-库塔(4,5)一步算法。对于多数问题求解,首选 ode45。
ode23	龙格-库塔(2,3)一步算法。对于计算结果精度要求较低及有轻微病态的微分方程,用 ode23 的效率比 ode45 高。
ode113	亚当斯多步算法。当要求高计算结果精度时,该算法效率比 ode45 高。
ode15s	专为解算病态系统而设计的基本预估-矫正法。但对于非病态系统,特别是当系统奇异或受到快变输入扰动时,该法不如其他方法有效。
ode23s	对计算结果精度要求较低的刚性微分方程,效果较 ode15s 好。
ode23t	内插梯形算法,适于有轻微刚性的方程。
ode23tb	隐性二阶算法。与 ode23s 一样,计算结果精度低时,效率较 ode15s 高。

(1) RelTol 控制计算过程的相对误差,通常在 0.1 和 1e-6 之间取值。RelTol 越小,积分的步数就越多,精度也越高。但是过小的 RelTol(如 1e-10),未必会给出很高的精度,原因是那时的计算截断误差将显著增加。

(2) 初始步长是仿真开始时所用的步长。在仿真开始后,算法会把算得的局部估计误差与设定的 RelTol 相比较,在满足 RelTol 的前提下,自动加大步长,提高计算效率。因此,产生一个数据输出点所取的步长通常不会小于最小步长。惟一的例外情况是离散采样周期小于初始步长。通常,初始步长都取得很小,如 1e-6。但要注意:如果初始步长设置过小,则当系统含间断点时,会产生大量的数据点,这会过于消耗计算机资源。

(3) 表 11-1 中的算法是变步长积分方法,但当最大步长等于最小步长时,仿

真将采用相应的定步长积分法。

11.3.4　离散系统的仿真

SIMULINK具有仿真多频采样离散系统和连续-离散混合系统的能力。

1. 离散模块

SIMULINK的每个离散模块都有一个内置输入采样器和输出零阶保持器。当离散模块和连续模块混用时，离散模块在采样间隔内的输出保持不变，而对离散模块的输入仅在采样的瞬间才被更新。

2. 采样周期

离散模块采样周期的设置是通过双击离散模块后所打开的对话框(图11.3.4-1)实现的。

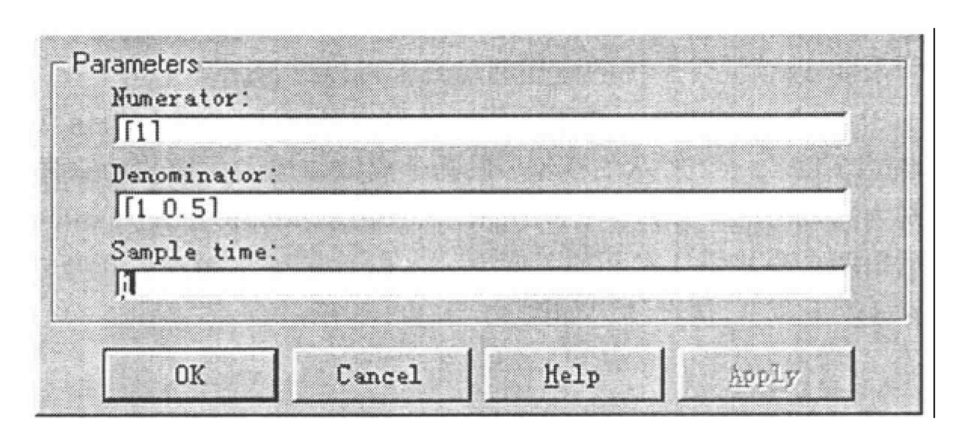


图11.3.4-1　离散传递函数模块的参数设置对话框

在Sample time栏里，若仅填写一个标量参数，那么它就是采样周期。若在此栏中填写二元向量，那么该向量的第一个元素指定采样间隔T_s，第二个元素设置偏移时间offset。实际采样时刻为：

$$t = n \times T_s + \text{offset}$$

式中，n为整数，offset是绝对值小于采样时间T_s的实数。若要求模型必须在某时刻更新，必须借助offset的设置来实现。

3. 纯离散系统

纯离散系统可使用任何一种积分算法进行仿真，而不会影响输出结果。若只要采样瞬间的输出数据，那么应把最小步长设置得比最大的采样间隔大。

4. 连续-离散混合系统

连续-离散混合系统由连续模块和离散模块混合组成。仿真时，可以采用前面介绍过的任何一种方法。但是对大多数混合系统来讲，变步长算法较好。

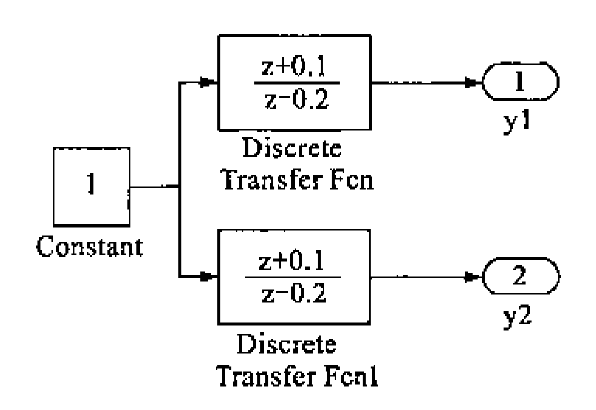


图 11.3.4-2　多采样速率的离散系统

5. 多频采样系统

多频采样系统包含不同采样速率的离散模块。在SIMULINK中，多频采样系统乃至多频采样-连续混合系统的建模和仿真都可以进行。

【例 11.3.4-1】　图 11.3.4-2 所示双速率采样系统 dsys.mdl 的仿真。

(1) 建立图 11.3.4-2 所示的模型。设两个离散传递函数模块的采样时间和偏移时间分别为[1,0.1]和[0.7,0]。Format 菜单中的 Sample Time Color 命令可以用颜色显示出两个模块采样时间的不同。

(2) 运行以下命令，仿真结果如图 11.3.4-3 所示。

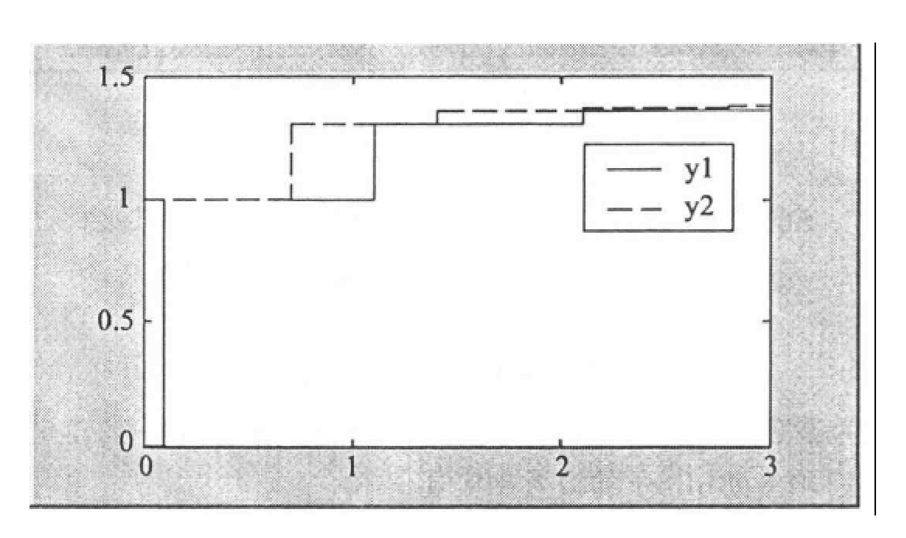


图 11.3.4-3　不同采样速率模块的输出结果

```
[t,x,y]= sim('dsys',3);
stairs(t,y)
legend('y(1)','y(2)')
```

说明：

①本例中阶跃输入函数的阶跃时刻设置为0；

②仿真所得系统输出y矩阵的第一列为模块1的输出，第二列为模块2的输出。

11.4 仿真系统的线性化模型

在非线性系统分析中，常需要在平衡工作点处提取线性模型。为此，SIMULINK提供了三个用于模型线性化的命令：linmod，dlinmod，trim。

11.4.1 连续系统的线性化模型

linmod和linmod2可以从SIMULINK所描写的动态系统(系统模型)提取一个用[A,B,C,D]表示的线性状态空间模型。它们的调用格式如下：

```
[A,B,C,D] = linfun('sys')
[A,B,C,D] = linfun('sys', x, u)
[A,B,C,D] = linfun('sys', x, u, pert)
[A,B,C,D] = linfun('sys', x, u, pert, xpert, upert)
```

说明：

(1) linfun linmod,dlinmod 或 linmod2。

(2) sys 欲提取线性化模型的被仿真系统文件名(mdl 文件)。

(3) x,u 分别为状态向量和输入向量。如果指定，则提取设定工作点的线性化模型。缺省值为零。

(4) pert 是全局扰动因子。缺省值为le-5。

(5) xport,uport 设置单个状态扰动因子和输入扰动因子的向量。如果指定，则忽略pert参数。

(6) linmod2所得模型比linmod准确，当然所需的运行时间也更多。linmod2会在圆整误差和截断误差之间取最好的折衷。

11.4.2 离散系统的线性化模型

dlinmod能够从非线性多频采样连续-离散混合系统中提取一个在任何给定的采样周期ts下的近似线性模型。当ts取零时，可得到近似的连续线性模型；否则，得到离散线性模型。该命令的一般调用格式是：

```
[Ad,Bd,Cd,Dd]= dlinmod('sys',ts,x,u,pert,xpert,upert)
```

说明：

(1) dlinmod只是在linfun中插入ts。ts指定采样周期。

(2) 在原系统稳定的前提下，若 ts 是原系统所有采样周期的整数倍，则由 dlinmod 所得线性模型在 ts 采样点上与原系统有相同的频率响应和时间响应。即使在上述条件不满足的情况下，该命令仍可能给出有效的线性化模型。

(3) Ad 的特征根在S平面的分布反映了系统的稳定性。当 ts=0 时，若 Ad 的所有特征根在左半平面，表示系统稳定；当 ts>0 时，Ad 特征根在单位圆内，则系统稳定。

(4) 如果原系统不稳定或 ts 不是原系统采样周期的整数倍，所得 Ad，Bd 有可能是复数阵。即使如此，Ad 的特征根仍能为稳定性判断提供一定的信息。

(5) dlinmod 可以把系统从一种采样周期模型变换到另一种采样周期下的模型，可以把离散模型变成连续模型，也可以把连续模型变成离散模型。

11.4.3 关于模型线性化的几点说明

1. 若被线性化系统本身是线性的，那么线性化过程不存在截断误差，扰动可取任意大小，工作点也不影响所得结果。一般说来，扰动取得较大，有利于减小圆整误差。

2. 在线性化过程中，模型的状态次序不变。

3. 在 SIMULINK 模型中，系统的输入输出要依次编号。

4. SIMULINK 模型的各模块与状态变量的对应关系可用 11.3.3 节【例 11.3.3-1】所讲的方法查询。

5. 所获得的线性模型，可以用控制工具箱、鲁棒控制工具箱及其他工具箱作进一步的处理。

11.4.4 确定平衡点

在给定输入、输出及状态条件下，系统平衡工作点的求取由 trim 命令实现。其调用格式：

```
[x,u,y,dx] = trim('sys',x0,u0,y0,ix,iu,iy,dx0,idx,options,t)
[x,u,y,dx,options] = trim('sys',...)
```

sys 是被研究的系统文件名。ix，iu，iy 都是整数向量，它们的元素指示初始猜测向量 x0，u0，y0 中哪些分量将固定不变。

除非问题本身的最小值惟一，否则不能保证所求得的平衡点是最佳值。因此，若想寻找全局最佳平衡点，那么多尝试几组初始猜测值是很有必要的。

当系统有不连续状态时，trim 一般不适用，而 trim4 也许能给出较好的结果。

【例 11.4.4-1】 求图 11.4.4-1 所示系统的线性模型、Bode 图及平衡工作点。

(1) 在 SIMULINK 建立如图 11.4.4-1 所示的模型(lin.mdl)。注意：作图中，

反馈模块输入输出口方向的改变可以用 Format 下拉菜单中的 Rotate Block 实现。

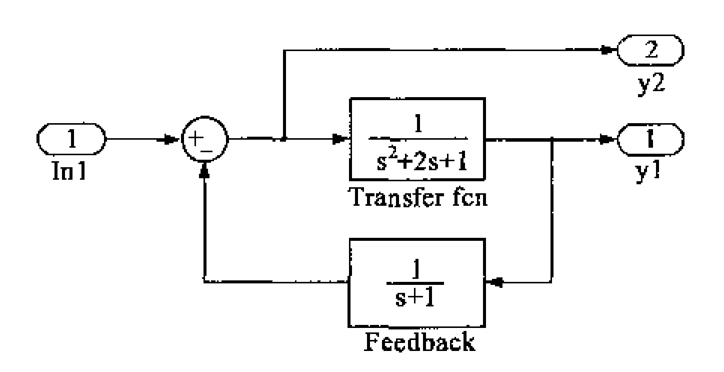


图 11.4.4-1　单输入双输出系统

（2）运行以下命令，可得到线性模型的[A，B，C，D]、传递函数及 Bode 图（见图 11.4.4-2）。

```
[A,B,C,D]=linmod('lin')
[num,den]=ss2tf(A,B,C,D);
printsys(num,den,'s')
bode(A,B,C,D)
A = -2     -1     -1
     1      0      0
     0      1     -1
B = 1
    0
    0
C = 0      1      0
    0      0     -1
D = 0
    1
num = 0          0.0000    1.0000    1.0000
      1.0000     3.0000    3.0000    1.0000
den = 1.0000     3.0000    3.0000    2.0000
                 3.1086e-015 s^2 + 1 s + 1
  num(1)/den = ---------------------------
                 s^3 + 3 s^2 + 3 s + 2
                 s^3 + 3 s^2 + 3 s + 1
  num(2)/den = ---------------------
```

```
s^3 + 3s^2 + 3s + 2
```

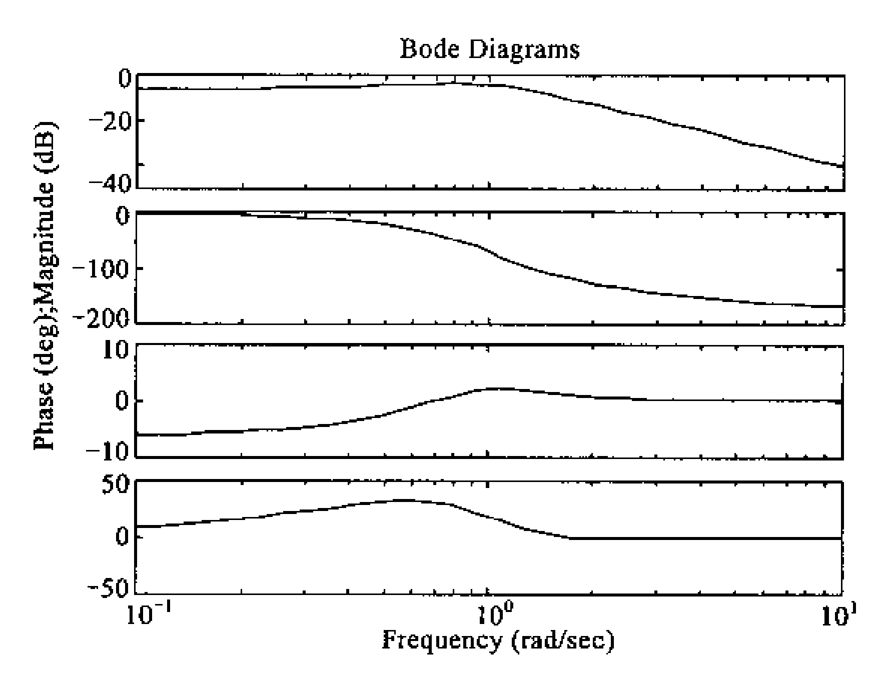


图 11.4.4-2　系统线性化模型的 Bode 图

(3) 运行以下命令求平衡点。

```
x=[0;0;0];
    u=0;
    y=[1;1];
    ix=[   ];              %不固定任何状态
    iu=[   ];              %不固定输入
      iy=[1:2];                %固定输出y(1)和输出y(2)
      [x,u,y,dx]= trim('lin',x,u,y,ix,iu,iy)
          x = 0.0000
              1.0000
              1.0000
          u = 2
          y = 1.0000
              1.0000
          dx =1.0e-015 *
              -0.2220
              -0.0227
              3331
```

11.5 S函数及其应用

S函数可以极大地扩展SIMULINK的功能。在SIMULINK环境中，若要充分发挥该软件的建模和仿真能力，应该了解和掌握S函数。引入S函数的目的是为了使SIMULINK有能力处理各种系统的仿真，如连续系统、离散系统、连续-离散混合系统、多频采样系统、嵌套系统等。可以用MATLAB或C编写S函数(C编写的S函数经编译后，成为MEX文件，方可在MATLAB中运行)，借助于S函数，用户可以在SIMULINK模型中加入自己编写的算法，使仿真过程更加真实准确地反映系统的动态过程。

11.5.1 S 函 数

S函数是SIMULINK用于描述动态系统的特殊语言结构。SIMULINK用特定的格式调用S函数，用户能方便地与SIMULINK的算法接口。

使用S函数优点在于：

(1) 可以通过S函数建立所需的线性、非线性模型；

(2) 将已有的C代码引入仿真模型；

(3) 可以编写所需的分析、仿真程序；

(4) 可以封装为一个通用模块，在建立仿真模型时，通过改变参数，作为不同的模块用于模型的建立。

编写S函数有特定的语法结构，用户要编写S函数，可以参照SIMULINK模块目录下的模板程序。如果用S函数去处理问题，SIMULINK将在复杂系统的分析和仿真中表现出很强的功能。对于用微分方程描述的动态系统来说，用M文件或C-MEX文件建立S函数比方框图建模具有更大的灵活性。S函数不管用什么方式创建，一旦建立，它既可以在框图中使用，也可以在命令中调用。

11.5.2 S函数的工作方式

对于M文件的S函数，SIMULINK通过传递参数flag，表示SIMULINK的当前状态，S函数根据flag进行不同的操作，因此S函数中要注意编写处理不同flag的代码(一般用switch语句)。SIMULINK直接调用C-MEX S函数，无需flag参数。表11-2是flag取不同数值时，S函数完成的功能。

表 11-2

Flag	S函数功能
0	返回系统变量和初始条件的维数
1	返回系统的状态导数 dx/dt
2	更新离散状态
3	返回系统的输出向量 y
4	更新下一个离散状态的时间间隔

S函数的调用格式是：

sys=sfunction(t,x,u,flag)

其中，sfunction是用户定义的系统，t是当前时刻，x是当前状态值，u是当前系统输入值，而变量flag的值控制返回变量sys的信息。

如果令flag=0，调用S函数：

[sys x0 str ts]= sfunction([ ],[ ],[ ],0)

那么返回参数x0表示状态向量的初始值，而返回参数sizes各分量的含义如下：

sizes(1)____连续状态变量数；

sizes(2)____离散状态变量数；

sizes(3)____输出变量数；

sizes(4)____输入变量数；

sizes(5)____系统有无代数循环的标志(有，则置1)；

sizes(6)____采样时间。

在运用S函数进行仿真运算时，必须清楚地知道系统不同时刻所需要的信息。例如开始进行仿真时，应先知道系统有多少状态变量，其中哪些是连续变量，哪些是离散变量，这些变量的初始条件等信息。而这些信息的获取，可由在S函数中设置flag=0获取。若系统是严格连续的(不含离散环节)，那么在每一仿真时步中需要知道的是给定时刻的系统状态导数(令flag=1可得)和系统输出(令flag=3可得)。若系统是严格离散的(不含连续环节)，令flag=2可以获得下一个离散状态，令flag=3获取离散系统的输出。

11.5.3　编写S函数

1. M文件S函数

M文件S函数编写格式与普通函数一样，只是对于不同的SIMULINK状态

标志，S 函数应有相应的例程，以完成相应操作。表 11-3 是一个 S 函数可能包含的例程。

表 11-3

S-函数例程	功能	状态标志
mdlInitializesizes	初始化。定义 S-函数参数，如采样时间，连续、离散状态的初始条件，变量数量。	flag=0
mdlDerivatives	计算连续状态变量的微分。	flag=1
mdlUpdate	更新离散状态变量、采样时间及主时间步长。	flag=2
mdlOutputs	计算 S 函数的输出。	flag=3
mdlGetTimeOfNextVarHit	若初始化时定义了变步长采样时间，此例程计算下一个采样间隔。	flag=4
mdlTerminate	终止仿真。	flag=9

编写 S 函数要考虑两个问题：

(1) 初始化。定义 S 函数参数，如采样时间，连续、离散状态的初始条件，变量数量。

(2) 编制表示系统特征及相应算法的例程。

为使 SIMULINK 能运行 S 函数，要初始化 S 函数，通知 SIMULINK 该 S 函数表示的系统的特征，如输入、输出的个数，连续和离散状态变量的初始条件，采样时间等。初始化由 mdlInitializesizes 完成，首先调用 simsizes：

```
sizes = simsizes;
```

simsizes 返回未初始化的 sizes 结构，接下的语句把系统的初始化参数填入该结构，表 11-4 描述了 sizes 结构各域的意义。初始化 sizes 结构后，再次调用

simsizes:

sys = simsizes(sizes);

传递初始化后的sizes结构给向量sys,sys保存着SIMULINK执行S函数所需的信息。

表11-4 sizes结构各域意义

域　　名	域名说明
Sizes. NumContStates	连续状态变量数
Sizes. NumDiscStates	离散状态变量数
Sizes. NumOutputs	输出数
Sizes. NumInputs	输入数
Sizes. DirFeedthrough	直接前馈标志(有1无0)
Sizes. NumSampleTimes	采样时间

【例11.5.3-1】 用M文件编写一个可变增益的S函数,其中形参n是增益。

```
function [sys,x0,str,ts] = timesN(t,x,u,flag,n)
switch flag, % 对于不同的flag,由开关语句控制执行不同的例程
case 0
[sys,x0,str,ts] = mdlInitializeSizes;                    % 初始化
case 3
sys = mdlOutputs(t,x,u,n);                          % 计算输出结果
case { 1, 2, 4, 9 }
sys = [ ];                          %若flag=1,2,4,9状态,返回空
otherwise
error(['Unhandled flag =',num2str(flag)]); % 对于其他flag值,调用出错处理
end;
```

SIMULINK传给S函数的前四个参数必须是t,x,u,flag,其中t为时间,x为状态向量,u为输入,flag为控制标志。返回值按sys,x0,str,ts的顺序排列,sys的内容取决于flag,如flag=0,sys是sizes结构;flag=3,sys是输出。flag=0时,x0为初始状态向量,flag为其他值时,忽略。str为保留,S函数中置为[]。ts为包含采样时间和偏移的n×2矩阵。

```
%==============================
% 初始化函数mdlInitializeSizes
%==============================
```

```
function [sys,x0,str,ts] = mdlInitializeSizes
sizes = simsizes;          % 调用函数simsizes,建立sizes 结构
% 给 sizes 结构赋值
sizes.NumContStates= 0;     % 没有连续状态变量
sizes.NumDiscStates = 0;    % 没有离散状态变量
sizes.NumOutputs= 1;        % 1 个输出
sizes.NumInputs= 1;         % 1 个输入
sizes.DirFeedthrough=1;     % 有直接前馈
sizes.NumSampleTimes=1;     % 采样时间
sys = simsizes(sizes);      % sizes 结构赋给向量 sys
x0 = [ ];                   % 没有连续状态变量
str = [ ];                  % 无状态序列
ts = [-1 0];                % 沿用前一个模块的采样时间
% End of mdlInitializeSizes
%==============================================
% 运算输出函数 mdlOutputs
%==============================================
function sys = mdlOutputs(t,x,u,n)
sys = n * u;
% End of mdlOutputs
```

2. C-MEX S 函数

C-MEX S 函数有着与M 文件S 函数同样的结构和函数,只是SIMULINK 并不传送 flag 给 C 函数。在不同的仿真阶段,SIMULINK 根据C 函数名调用C 函数,因此C-MEX S 函数一定要有与 M 文件的例程(表 11-3)同名的 C 函数,否则SIMULINK 不能正确调用C-MEX S 函数。另外,所有函数应以 mdl 打头。下面介绍 C-S 函数的一般格式。

【例 11.5.3-2】 用C 编写【例 11.5.3-1】的 S 函数。

(1) S 函数头部格式

```
#define S_FUNCTION_NAME timesN
#define S_FUNCTION_LEVEL 2
#include "simstruc.h"
```

函数头部定义了函数名,包含文件 simstruc.h。SIMULINK 通过结构 SimStruct 与S 函数交换数据,该结构在头文件 simstruc.h 中定义。第二条语句是版本标识。

(2) 函数结尾

```
#ifdef MATLAB_MEX_FILE /* Is this file being compiled as a MEX-file? */
```

```
#include "simulink.c" /* MEX-file interface mechanism */
#else
#include "cg_sfun.h" /* Code generation registration function */
#endif
```

根据不同的应用包含不同的文件，如要编译成 MEX 文件，包含 simulink.c；如要与 Real-Time Workshop 连接，编译成可实时、独立运行的程序，则包含文件 cg_sfun.h。

(3) 其他调用函数

```
static void mdlInitializeSizes(SimStruct *S)
{
ssSetNumSFcnParams(S, 0);
if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S))
{
return; /* Parameter mismatch will be reported by Simulink */
}
if (! ssSetNumInputPorts(S, 1))
  return;
ssSetInputPortWidth(S, 0, DYNAMICALLY_SIZED);
ssSetInputPortDirectFeedThrough(S, 0, 1);
if (! ssSetNumOutputPorts(S,1))
  return;
ssSetOutputPortWidth(S, 0, DYNAMICALLY_SIZED);
ssSetNumSampleTimes(S, 1);
/* Take care when specifying exception free code - see sfuntmpl.doc */
ssSetOptions(S, SS_OPTION_EXCEPTION_FREE_CODE);
}
static void mdlInitializeSampleTimes(SimStruct *S)
{
  ssSetSampleTime(S, 0, INHERITED_SAMPLE_TIME);
  ssSetOffsetTime(S, 0, 0.0);
}
static void mdlOutputs(SimStruct *S, int_T tid) //计算输出
{
  int_T i;
  InputRealPtrsType uPtrs = ssGetInputPortRealSignalPtrs(S,0);
  real_T *y = ssGetOutputPortRealSignal(S,0);
  int_T width = ssGetOutputPortWidth(S,0);
```

```
    for (i=0; i<width; i++)
    {
        *y++ = 2.0 *(*uPtrs[i]);
    }
}
static void mdlTerminate(SimStruct *S){} //终止仿真
```

说明：

(1) mdlInitializeSizes 是 SIMULINK 调用的第一个函数，它完成对 S 函数的初始化。初始化过程依靠一系列宏来完成，如程序中宏 ssSetNumSFcnParams(S, 0)完成对 S 函数参数个数的设置。

(2) 开头和结尾的语句以及初始化都是必需的，其他函数可根据 S 函数的具体功能编写。

(3) C 语言编写的 S 函数，有确定的函数名，不能任意命名函数名。

有关编写 M 文件和 C-MEX 的 S 函数的详细语法可参考 MATLAB 的帮助文件 sfunctions.pdf。MATLAB 的 SIMULINK 工具箱有很多用 MATLAB、C 语言编写的范例，可以用作参考。C-MEX 比 M 文件更灵活，实现的功能更多。一般情况下，C-MEX 比 M 文件运行速度快，如果同时存在同名的两种类型文件，SIMULINK 优先采用 C-MEX 的 S 函数。

11.5.4 在 SIMULINK 中引用 S 函数

在 SIMULINK 中建立仿真模型时，可以用 Function&Tables 库中的 S-Function 模块引用 S 函数，成为用户自定义的 SIMULINK 模块，这样可以灵活地利用自编的 S 函数建立仿真模型。下面介绍 S 函数在 SIMULINK 中的引用方法。

【例 11.5.4-1】 用 M 文件编写一个限幅器的 S 函数，并用 S-Function 模块在仿真模型中调用它。限幅器的数学模型：

$$x = \begin{cases} 0 & (x \leqslant lb, u < 0) \text{ 或} (x >= ub, u > 0) \\ u & \text{其他} \end{cases}$$

其中，x 为状态，u 是输入，lb 和 ub 表示限幅的下限和上限。

(1) 根据数学模型，编写 S 函数。

```
function [sys,x0,str,ts]=limintm(t,x,u,flag,lb,ub,xi)
switch flag
  case 0
    [sys,x0,str,ts] = mdlInitializeSizes(lb,ub,xi);
  case 1
    sys = mdlDerivatives(t,x,u,lb,ub);
```

```
    case {2,9}
      sys = [];
    case 3
      sys = mdlOutputs(t,x,u);
    otherwise
      error(['unhandled flag =',num2str(flag)]);
  end
  function [sys,x0,str,ts] = mdlInitializeSizes(lb,ub,xi)
  sizes = simsizes;
  sizes.NumContStates  = 1;
  sizes.NumDiscStates  = 0;
  sizes.NumOutputs     = 1;
  sizes.NumInputs      = 1;
  sizes.DirFeedthrough = 0;
  sizes.NumSampleTimes = 1;
  sys = simsizes(sizes);
  str = [];
  x0 = xi;
  ts = [0 0];     % sample time: [period, offset]
  function sys = mdlDerivatives(t,x,u,lb,ub)
  if (x <= lb & u < 0) | (x>= ub & u>0 )
    sys = 0;
  else
    sys = u;
  end
  function sys = mdlOutputs(t,x,u)
  sys = x;
```

（2）从函数及表格库（Functions&Tables）选取 S-Function 模块，再选 Sine Wave 和 Scope，建立仿真模型（图 11.5.4-1）。

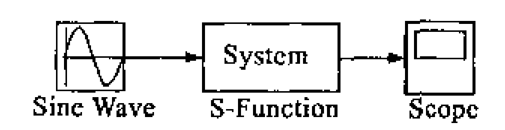


图 11.5.4-1　限幅器仿真模型

（3）引用 limintm S 函数。双击 S-Function 模块图标，弹出对话框（图 11.5.4-2），在 S-function name 栏填入 S 函数文件名 limintm，在 S-function parameters 栏

填入要传送的三个变量lb,ub,xi,单击OK。S-Function函数模块图标名system改为limintm。

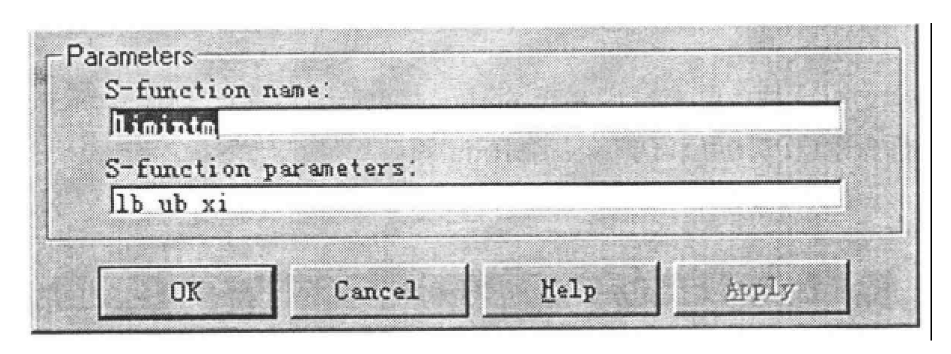


图11.5.4-2 S-Function对话框

(4) 在MATLAB窗口输入lb=0,ub=1,xi=0,切到SIMULINK窗口仿真,图11.5.4-3是仿真结果。

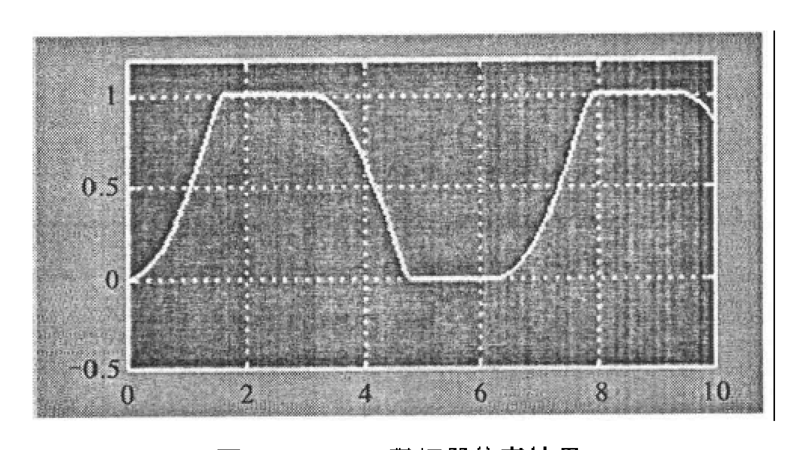


图11.5.4-3 限幅器仿真结果

11.5.5 S函数文件转化为SIMULINK模块

前面介绍了利用S-Function模块将limintm.m引入SIMULINK模型,由于有附加参数lb,ub及xi,因此仿真时要与MATLAB工作空间进行数据交换,即须先在MATLAB工作间中定义三个参数lb,ub和xi,否则仿真不能正常进行。另外,每次使用S函数都要借助S-Function模块,也很不方便。SIMULINK提供的封装(Masking)功能允许用户改变模块(包括子系统和S函数模块)的属性。通过Masking封装M文件和MEX文件的S函数,可以把任何描述动态系统的S函数构造成标准的SIMULINK模块,这样使用起来更方便。下面介绍如何把上节的限

幅器的 S 函数 limintm. m 封装成 SIMULINK 模块。

【例 11. 5. 5-1】 把限幅器的 S 函数 limintm 封装成 SIMULINK 模块。

(1) 改写模块名称。在图 11. 5. 4-1 所示系统中，单击 S-Function 模块的标题，并将其改为“Limitr”。

(2) 建立对话框。单击选中 S-Function 模块，在 Edit 菜单中选择 Edit Mask，弹出对话框(图 11. 5. 5-1)，选 Initialization 页。

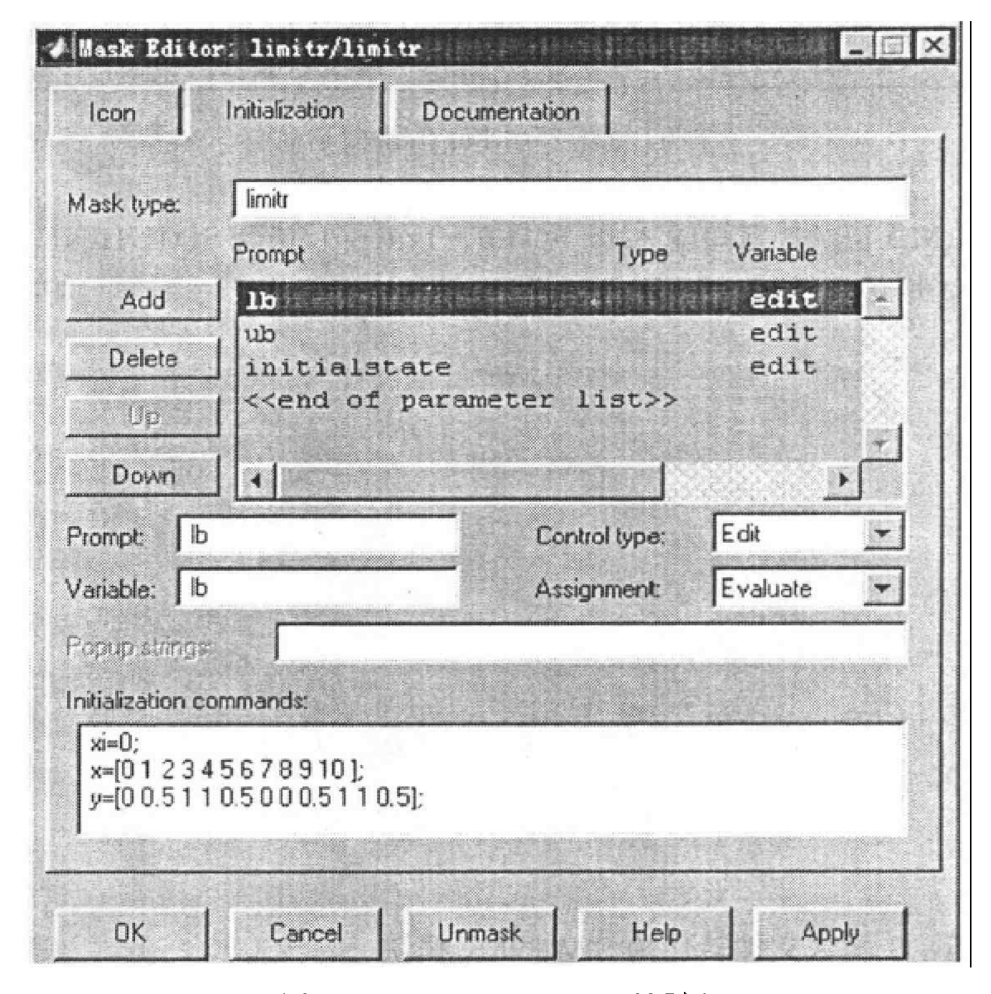


图 11. 5. 5-1　Mask Edit 对话框

(a) 给模块命名。在 Mask type 栏输入模块名“limitr”。

(b) 设置参数对话框。单击 Add，旁边的提示栏增加一行(蓝色加亮)，在 Prompt 编辑框输入提示标题 lower bound，Variable 编辑框输入变量名 lb。Control type 是变量输入的方式(分别是 Edit——编辑框，Checkbox——复选框，Popup——下拉菜单)，这里选 Edit。Assignment 是变量的赋值方式，Evaluate 表示将输入求值后赋给变量，Literal 表示直接将输入赋给变量，例中选 Evaluate 赋

值方式。

(c) Initialization commands 定义 mask 工作空间的变量，SIMULINK 初始化模块，绘制模块图标都要用到这些变量。Initialization commands 可以是 MATLAB 表达式、函数以及在 mask 工作空间定义的变量。为避免在 MATLAB 窗口显示执行结果，初始化命令用分号结尾。

(3) Icon 页，定义模块外观。

(a) Drawing commands 可以在 Drawing commands 栏输入 MATLAB 命令，定义模块图标、描述模块功能、模块所表示的传递函数以及状态方程等。如输入命令 disp('limitr')，将在图标上显示模块名 limitr；若显示内容较多，需分行显示，在字中中加换行符\n。图 11.5.5-2(a)是输入

disp('limitr\nx=0 (x<=lb & u<0)\n x=u (x>=ub & u>0)')

的结果。

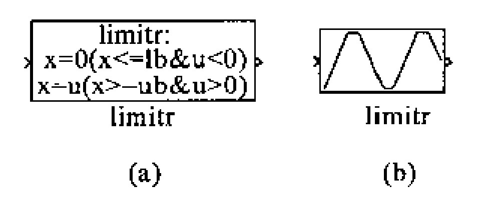


图 11.5.5-2 显示模块描述

在 Initialization 页 Initialization commands 栏定义两个向量 x，y：

x=[0 1 2 3 4 5 6 7 8 9 10]；

y=[0 0.5 1 1 0.5 0 0 0.5 1 1 0.5]；

在 Drawing commands 中输入：

plot(x,y)

模块外观如图 11.5.5-2(b)。

(b) 模块显示方式 Icon 页有四个控制模块外观的参数。Icon frame 控制模块图标外框的显示；Icon transparency 表示是否显示被覆盖的模块信息。如本例封装 S 函数 limintm ，如果选 tranparent，图标会显示 limintm(图 11.5.5-3a)；Icon rotation 若选择 rotates，当模块旋转时，模块图标上的图形跟着一起旋转(图 11.5.5-3 b)，若选择 fixed，图形不动(图 11.5.5-3 c)；Drawing coordinates 定义绘图坐标系统。Autoscale-x，y 向量分别表示图标横标和纵标，图标左下角是 min x，min y，右上角是 max x，max y。Normalized——规格化坐标，图标左下角是(0,0)，右上角为(1,1)。Pixel——用像素表示 x，y 的值，图标左下角为(0,0)，右上角是图标的宽和高。

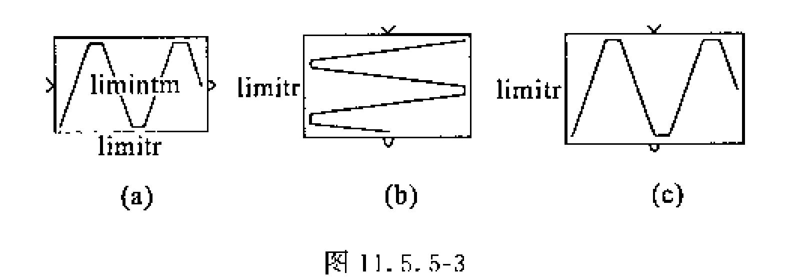


图 11.5.5-3

（4）在 Documentation 页的 Block discription 输入描述模块功能的文字，Block helps 编制帮助文件。当在SIMULINK 模型窗口双击模块图标时，弹出的参数对话框将在模块说明处显示 Block discription，单击 Help，显示 Block helps 的内容。

像其他标准模块一样，这个封装后的模块具有独特的图标和方便易用的对话框。通过"封装"技术，用户可以建立自己的SIMULINK 模块、模块库。如果要更改模块的属性，可用 Edit Mask 修改。若要解除"封装"，单击对话框的 Unmask。

11.5.6 创建子系统

SIMULINK 还提供创建子系统（Subsystem）的功能，子系统的建立有利于管理大型系统。当一个动态模型包含较多模块时，最好把系统按功能分块，建立子系统，这样便于仿真的调试、排错。实际上，SIMULINK 模块库中的许多模块（如PID）本身就是由多个模块构成的子系统。下面讨论如何建立子系统。

有两种方法建立子系统：

1. 用 subsystem 模块

（1）从 Signals & Systems 库复制 subsystem 模块到正在建立的模型中（图 11.5.5-4）。

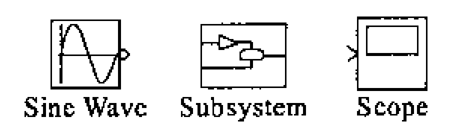


图 11.5.5-4 复制 subsystem 模块

（2）双击 subsystem 模块，打开它（新窗口 untitled/subsystem）。

（3）在新窗口 untitled/subsystem 创建子系统（图 11.5.5-5），注意子系统用 import 模块作为输入端口，用 Outport 模块作为输出端口。

（4）关闭 untitled/subsystem 窗口，含有子系统的模型如图 11.5.5-6。

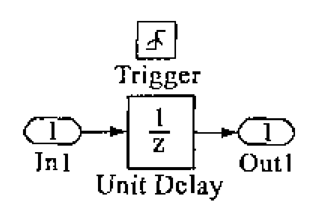


图 11.5.5-5 在 subsystem 中建立子系统

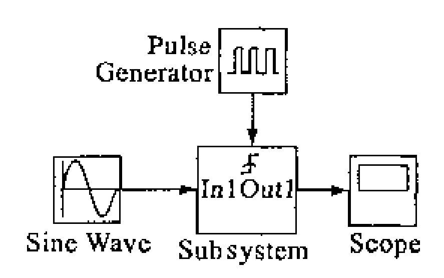


图 11.5.5-6 含有子系统的模型

2. 用菜单命令 Create Subsystem

如果想把模型中的某些模块合并成子系统，可用此法。

(1) 选择子系统的模块。按下鼠标左键并拖动鼠标，用虚线框围住子系统要包含的模块及连线，松开左键，所有被选中的元素会显示黑块控点(图 11.5.5-7 a)。

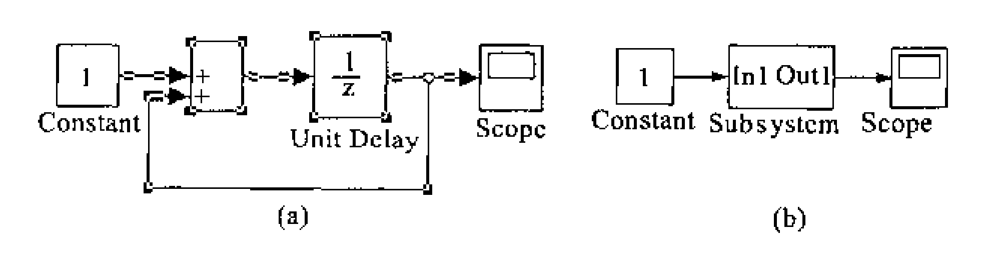


图 11.5.5-7

(2) 选择 Edit 菜单下的 Create Subsystem，SIMULINK 将所有选中的模块转化为一个子系统(图 11.5.5-7b)。双击 subsystem 模块可以看到该子系统的构成。单击标题 Subsystem，可以把 Subsystem 换成能够比较清楚表示子系统功能的名字。

习　题　十一

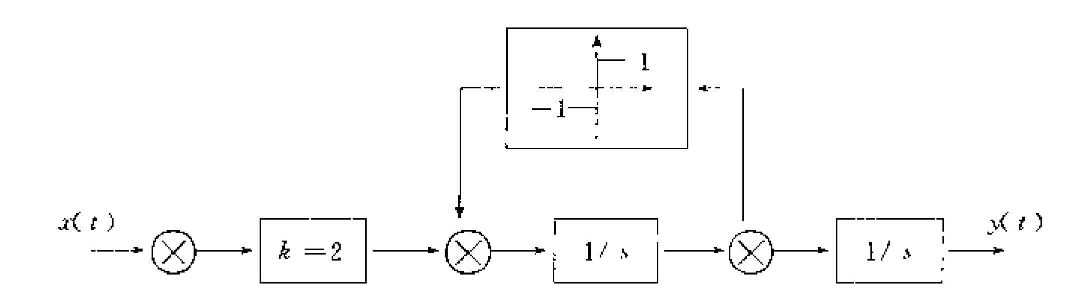
1．对不同触发方式(上升沿触发，下降沿触发，双边触发)的触发器进行仿真，比较不同触发方式的差异。要求在同一个示波器上显示。

2．设计一个滤波器的simulink模型，仿真其对含有两个正弦波(信号100Hz，振幅150；噪音25Hz，振幅30)的信号进行滤波。

3．有一数字系统，输入时钟信号频率1Hz，输出是8分频的脉冲，脉冲宽度与时钟信号宽度一样，构造其simulink模型。考虑输出8倍频脉冲又该如何设计。

4．仿真小球回弹实验。小球弹性系数0.8，初速15m/s，高度10m。要求输出速度、高度。

5．构造下面框图的simulink模型，对其仿真，输出不同阶跃输入的响应图形。



第十二章　机械振动的仿真

12.1 概　　述

机械振动是工程中常见的物理现象。悬挂在弹簧上的物体在外界干扰下所作的往复运动就是最简单直观的机械振动。广泛地说，各种机器设备及其零部件和基础，都可以看成是不同程度的弹性系统。例如桥梁在车辆通过时引起的振动，汽轮机、发电机由于转子不平衡引起的振动等。因此，机械振动就是在一定的条件下，振动体在其平衡位置附近所作的往复性的机械运动。

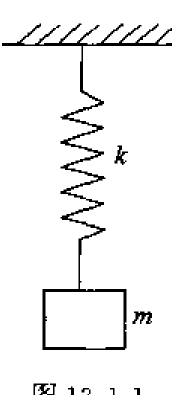


图 12.1-1

实际中的振动系统是很复杂的。为了便于分析研究和使用数学工具进行计算，需要在满足工况要求的条件下，把实际的振动系统简化为力学模型。例如图 12.1-1 所示就是个最简单的单自由度质量(m)-弹簧(k)系统。

如果实际系统很复杂，要求的精度较高，简化的力学模型也就复杂，即形成多自由度振动系统。

振动系统中各参数的动态特性，可以用常系数线性微分方程来描述的，称为线性振动。但工程实际中也有很多振动系统是不能线性化的，如果勉强线性化，就会使系统的性质改变，所得的系统只能按非线性振动系统处理。

机械振动分析方法很多。对于简单的振动系统，可以直接求解其微分方程的通解。由于计算机进行数值计算非常方便，所以振动的微机仿真是一种最直接的方法。

由于振动模型中尤其是多自由度振动很方便用矩阵微分方程来描述，所以 MATLAB 语言在振动仿真中体现出十分优越的特性。

本章先介绍机械振动的单自由度、多自由度振动的基础，然后介绍仿真计算的各种计算公式，最后通过 MATLAB 语言来实现。

12.2 单自由度系统的振动

12.2.1 无阻尼自由振动

图12.1-1所示的单自由度振动系统可以用如下微分方程描述：

$$m\ddot{x} + kx = 0 \qquad (12.2.1\text{-}1)$$

令 $\omega_n^2=\frac{k}{m}$，方程的通解为

$$x = a\sin\omega_n t + b\cos\omega_n t \qquad (12.2.1\text{-}2)$$

式(12.2.1-2)表示了图12.1-1中质量 m 的位置随时间而变化的函数关系，反映了振动的形式与特点，称为振动函数。

式(12.2.1-2)中，a，b 为积分常数，它决定于振动的初始条件。如假定 $t=0$ 时，质量块的位移 $x=x_0$，其速度 $\dot{x}=\dot{x}_0=V_0$，则

$$a = \frac{V_0}{\omega_n},\ b = x_0$$

即

$$x = \frac{V_0}{\omega_n}\sin\omega_n t + x_0\cos\omega_n t \qquad (12.2.1\text{-}3)$$

或写成

$$x = A\sin(\omega_n t + \varphi) \qquad (12.2.1\text{-}4)$$

$$A = \sqrt{(\frac{V_0}{\omega_n})^2 + x_0{}^2},\ \varphi = a\tan\frac{x_0\omega_n}{V_0}$$

其中 A 为振幅，ω_n 为振动圆频率，φ 为相位角，$f_n=\omega_n/2\pi$(赫兹)称为固有频率。固有频率与外界给予的初始条件无关，它是系统本身所具有的一种重要特性。

12.2.2 有阻尼自由振动

图12.1-1所示的自由振动中，由于系统的能量守恒，如果振动一旦发生，它就会持久的、等幅的一直进行下去。但是，实际上所遇到的自由振动都是逐渐衰减直至最终停止的，即系统存在阻尼。阻尼有相对运动表面的摩擦力、液体与气体的介质阻力、电磁阻力以及材料变形时的内阻力等。

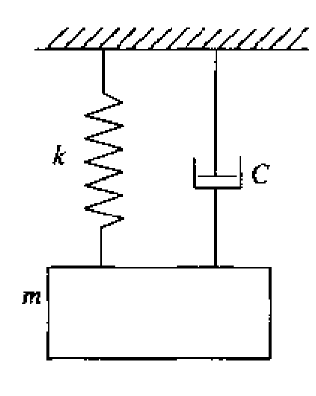


图12.2.2-1

图12.2.2-1所示为考虑了阻尼的单自由度振动系统

模型。其运动微分方程为

$$m\ddot{x}+c\dot{x}+kx=0 \quad (12.2.2\text{-}1)$$

令

$$\frac{c}{m}=2n,\ \frac{k}{m}=\omega_n^2$$

则

$$\ddot{x}+2n\dot{x}+\omega_n^2x=0 \quad (12.2.2\text{-}2)$$

其通解为

$$x=e^{-nt}(c_1e^{\sqrt{n^2-\omega_n^2}}+c_2e^{-\sqrt{n^2-\omega_n^2}}) \quad (12.2.2\text{-}3)$$

式中 c_1,c_2 为积分常数,由振动初始条件确定。

令$\frac{n}{\omega_n}=\xi$,称为相对阻尼系数或阻尼率。则式(12.2.2-3)可写为

$$x=e^{-\xi\omega_nt}(c_1e^{\omega_nt\sqrt{\xi^2-1}}+c_2e^{-\omega_nt\sqrt{\xi^2-1}}) \quad (12.2.2\text{-}4)$$

由此可以讨论阻尼对系统的自由振动将会产生的影响。

一、当 $\xi<1$ 时,称为弱阻尼状态

此时,ξ^2-1 为虚数,式(12.2.2-4) 变为

$$x=e^{-\xi\omega_nt}(c_1e^{i\sqrt{1-\xi^2}\omega_nt}+c_2e^{-i\sqrt{1-\xi^2}\omega_nt}) \quad (12.2.2\text{-}5)$$

利用欧拉公式,式(12.2.2-5)可写为

$$x=Ae^{-\xi\omega_nt}[b\cos\sqrt{1-\xi^2}\omega_nt+a\sin\sqrt{1-\xi^2}\omega_nt] \quad (12.2.2\text{-}6)$$

括号内为两个简谐振动相加,即式(12.2.2-6)可写为

$$x=Ae^{-\xi\omega_nt}\sin(\sqrt{1-\xi^2}\omega_nt+\varphi) \quad (12.2.2\text{-}7)$$

$$A=\sqrt{\frac{(V_0+\xi\omega_2x_0)^2+x_0\omega_n\sqrt{1-\xi^2}}{\omega_n^2(1-\xi^2)}},\varphi=\arctan(\frac{x_0\omega_n\sqrt{1-\xi^2}}{V_0+\xi\omega_nx_0})$$

由式(12.2.2-7)可以看出,这种弱阻尼自由振动的几种特性:

1. 它是一个简谐振动,振动的频率为$\sqrt{1-\xi^2}\omega_n$,这是 ω_n 为无阻尼时系统的固有频率。一般情况下,ξ 常在 0.1 左右,因此对固有频率的影响不大,即认为$\sqrt{1-\xi^2}\omega_n\approx\omega_n$。

2. 振动的振幅为 $Ae^{-\xi\omega_nt}$,其中 A,ξ,ω_n 皆为定值。所以振幅随时间变化的规律是一条指数递减曲线(图 12.2.2-2)。

二、当 $\xi>1$ 时,称为强阻尼状态

此时,式(12.2.2-4)可写成

$$x=c_1e^{(-\xi+\sqrt{\xi^2-1})\omega_nt}+c_2e^{(-\xi-\sqrt{\xi^2-1})\omega_nt} \quad (12.2.2\text{-}8)$$

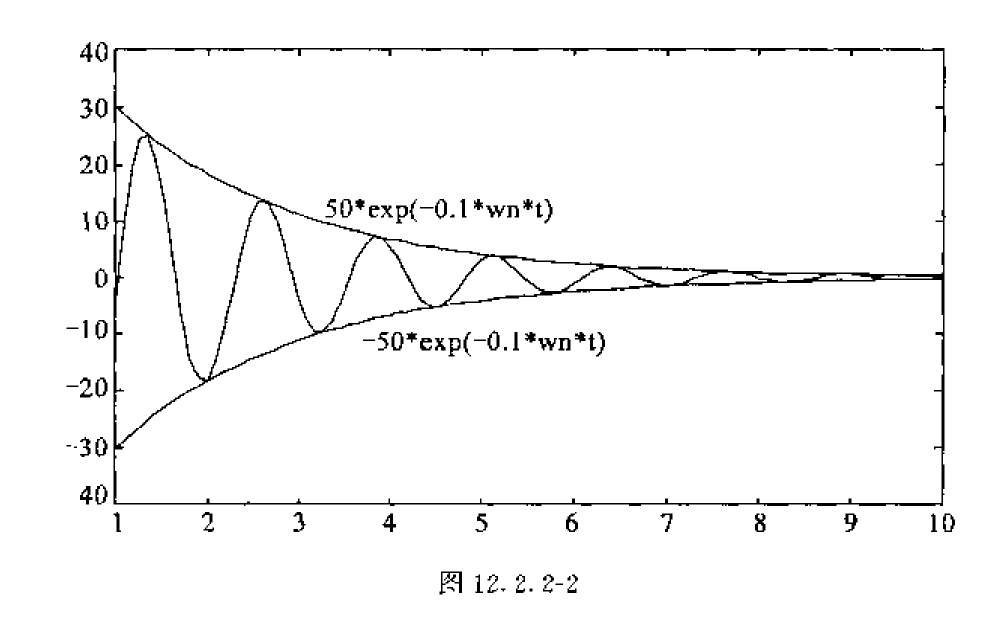


图 12.2.2-2

$$c_1=\frac{V_0+(\xi+\sqrt{\xi^2-1})\omega_n x_0}{2\omega_n\sqrt{\xi^2-1}},\ c_2=\frac{-V_0+(-\xi+\sqrt{\xi^2-1})\omega_n x_0}{2\omega_n\sqrt{\xi^2-1}}$$

由于$\xi^2-1>0$，故式(12.2.2-8)中二项指数皆为实数。又因为$\xi>\sqrt{\xi^2-1}$，故二项指数皆为负值，所以式(12.2.2-8)所表示的是一根指数递减曲线。这表示系统将不再产生前面所述的振动，而是产生一按指数规律衰减的蠕动(图 12.2.2-3)。

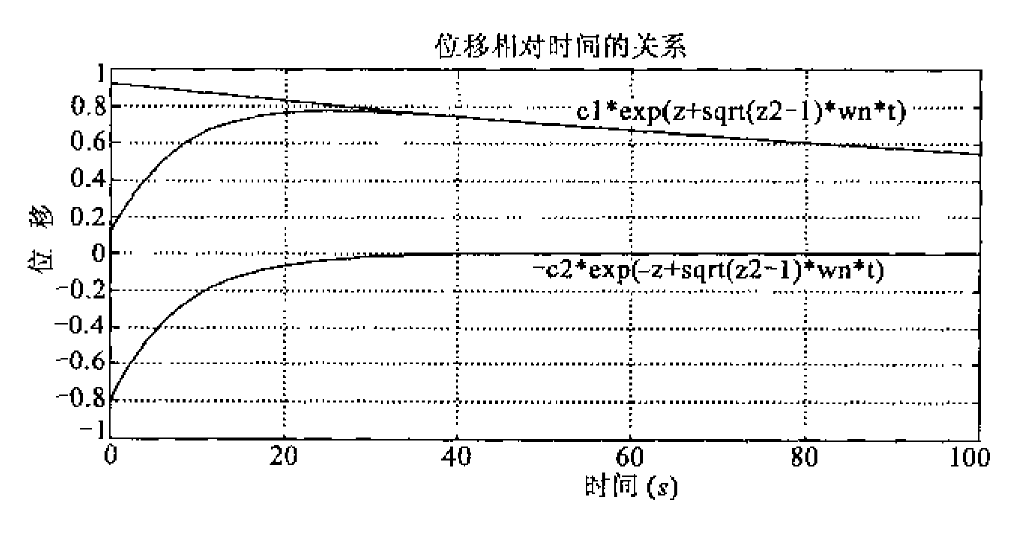


图 12.2.2-3

三、当$\xi=1$时，称为临界阻尼状态

由于 $\xi=\frac{n}{\omega_n}=1$，$n=\omega_n$则有

$$c_c = 2m\omega_n = 2m\sqrt{\frac{k}{m}} = 2\sqrt{km} \tag{12.2.2-9}$$

这里 c_c 为临界阻尼状态下的阻尼系数，称为临界阻尼系数，显然它是系统本身所具有的特性之一。

由 $\xi=\frac{n}{\omega_n}=\frac{c}{2m\omega_n}$ 及 $c_c=2m\omega_n$，有 $\xi=\frac{c}{c_c}$。也就是说，相对阻尼系数(阻尼率)ξ 反映了系统的实际阻尼与临界阻尼的关系。

在临界阻尼状态下，有

$$x = e^{-\omega_n t}(c_1 + c_2 t) \tag{12.2.2-10}$$

其中 $c_1=x_0$，$c_2=V_0+\omega_n x_0$。显然，在这种状态下不能形成振动(图 12.2.2-3)。

12.2.3　有阻尼自由振动响应计算与 MATLAB 实现

根据式(12.2.2-7)、(12.2.2-8)、(12.2.2-10) 编写的程序如下：

```
function vtb1(m,c,k,x0,v0,tf)
%vtb1 用来计算单自由度有阻尼自由振动系统的响应
%程序中设置了 6 个变量(即 m,c,k,x0,v0,tf)输入和 5 个变量(即 zeta,wn,x0,v0,tf)输入
%m 为质量;c 为阻尼;k 为刚度;x0 为初始位移;v0 为初始速度
%tf 为仿真时间
%zeta 为阻尼系数;wn 为固有频率
%程序中 z 为阻尼系数;A 为振动幅度;phi 为初相位
clc
%该循环确定输入方式是 VTB1(m,c,k,x0,v0,tf),还是 VTB1(zeta,w,x0,v0,tf)
if  nargin==5
    z=m;wn=c;tf=v0;v0=x0;x0=k;m=1;c=2*z*w;k=w^2;
end
wn=sqrt(k/m);%固有频率。
z=c/2/m/wn;
wd=wn*sqrt(1-z^2);
fprintf('固有频率为%.3g.rad/s.\n',wn);
fprintf('阻尼系数%.3g.\n',z);
fprintf('有阻尼的固有频率%.3g.\n',wd);
t=0:tf/1000:tf;
if  z<1
```

```
        A= sqrt(((v0+z*wn*x0)^2+(x0*wd)^2)/wd^2);
        phi =atan2(x0*wd,v0+z*wn*x0);
        x=A*exp(-z*wn*t).*sin(wd*t+phi);
        fprintf('A=%.3g\n',A);
        fprintf('phi=%.3g\n',phi);
    else if z==1
            a1=x0;
            a2=v0+wn*x0;
            fprintf('a1=%.3g\n',a1);
            fprintf('a2=%.3g\n',a2);
            x=(a1+a2*t).*exp(-wn*t);
    else
            a1=(-v0+(-z+sqrt(z^2-1))*wn*x0)*wn*x0)/2/wn/sqrt
            (z^2-1);
            a2=(v0+(z+sqrt(z^2-1))*wn*x0)/2/wn/sqrt(z^2 1);
            fprintf('a1=%.3g\n',a1);
            fprintf('a2=%.3g\n',a2);
            x=exp(-*wn*t).*(a1*exp(wn*sqrt(z^2-)*t)
                +a2*exp(wn*sqrt(z^2-1)*t));
    end
    plot(t,x),grid
    xlabel('时间(s)')
    ylabel('位移')
    title('位移相对时间的关系')
```

运行该程序时，只需要给出相应参数，例如

vtb1(1,0.05,1,1,1,100)

则显示固有频率为1(rad/s)，幅值为A=1.43，相位角为phi=0.773。其响应曲线如图12.2.3-1所示。

程序中if语句就是判断ξ大小的，即判断是弱阻尼状态、强阻尼状态还是临界阻尼状态。

如果要想求出振动的速度$\dot{x}$(=xd)和加速度$\ddot{x}$(=xdd)，只要对式(12.2.2-7)、式(12.2.2-8)、式(12.2.2-10)分别进行求导，在程序中加入相应的内容，最后增加plot(t,xd),plot(t,xdd)即可给出速度和加速度图。

12.2.4 有阻尼受迫振动

单自由度有阻尼强迫振动的微分方程为

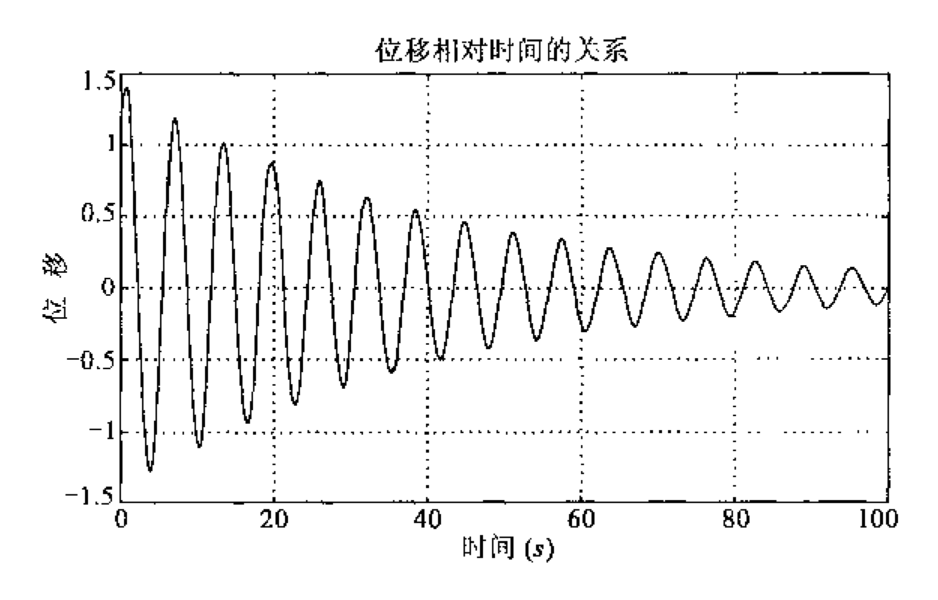


图 12.2.3-1

$$m\ddot{x} + c\dot{x} + kx = f(t) \tag{12.2.4-1}$$

式中 $f(t)$ 为外加的激励力。如果 $f(t)=F_0\sin\omega t$，则称为谐激励力，式(12.2.4-1)可写成

$$m\ddot{x} + c\dot{x} + kx = F_0\sin\omega t \tag{12.2.4-2}$$

式(12.2.1-2) 是一个线性非齐次方程。其振动响应为

$$\left.\begin{aligned} x &= Ae^{-\xi\omega_n t}\sin(\sqrt{1-\xi^2}\omega_n t + \varphi) + B\sin(\omega_n t - \varphi) \\ \varphi &= \arctan\frac{2\xi\lambda}{1-\lambda^2} \\ \lambda &= \frac{\omega}{\omega_n} \\ B &= \frac{\omega_n^2 F_0/k}{\sqrt{(\omega_n^2-\omega^2)^2+(2\xi\omega_n\omega)^2}} = \frac{F_0}{k\sqrt{(1-\lambda^2)^2+(2\lambda\xi)^2}} \end{aligned}\right\} \tag{12.2.4-3}$$

式中 A 与 φ 仍按式(12.2.2-7)计算，λ 为频率比，B 为稳态响应的振幅。

谐迫振动的主要特性有：

1. 式(12.2.4-3)包括瞬态与稳态响应两部分，其中瞬态响应是一个有阻尼的谐振。振动频率为系统固有频率 ω_n，振幅 A 与初相位角 φ 决定于初始条件，振幅的衰减按 $e^{-\xi\omega_n t}$ 规律，因此，振动持续时间决定于系统的阻尼比 ξ。

2. 谐振的稳态响应也是一个简谐振动，其频率比等于激励力的频率 ω，振幅

为 B，相位角为 φ。

3. F_0/k 是系统的静载荷 F_0 作用下产生的变形，称"静变位"。而系统在 $f(t)=F_0\sin\omega t$ 作用时，产生等幅振动，这个振动的振幅实质上是一种"动态变位"。$H(\omega)=B/(F_0/k)$ 即为"动态变位"与静态变形之比，称为动力放大因子。$H(\omega)$ 随阻尼比 ξ 和频率比 λ 而变化。当 $\lambda<<1$ 时，$H(\omega)\approx1$ 即 $B\approx F_0/k$，说明激励频率 ω 远小于系统固有频率 ω_n 时，系统可视为静态，振幅也等于静变位。当 $\lambda>>1$ 时，$H(\omega)\to0$ 即 $B\to0$，这是因为激励力频率非常高，系统由于惯性而来不及随之振动。当 $\lambda\approx1$ 时，B 急剧增大，即发生共振。

下面是单自由度谐迫振动计算程序。

```
function vtb2(m,c,k,x0,v0,tf,w,f0)
%单自由度系统的谐迫振动
wn=sqrt(k/m);
z=c/2/m/wn; %阻尼比
lan=w/wn; %频率比
wd=wn * sqrt(1-z ^ 2);
A=sqrt(((v0+z * wn * x0) ^ 2+ (x0 * wd) ^ 2)/wd ^ 2);
t=0:tf/1000:tf;
phi=atan2(2 * z * lam,1  lam ^ 2) %相位角
B=wn ^ 2 * f0/k/sqrt((wn ^ 2 -w ^ 2) ^ 2+(2 * z * wn * w) ^ 2);
x=A * exp(-z * wn * t). * sin(sqrt(1-z ^ 2) * wn * t+phi)+B * sin(w * t+phi);
plot(t,x),grid
xlabel('时间(s)')
ylabel('位移')
title('位移与时间的关系')
```

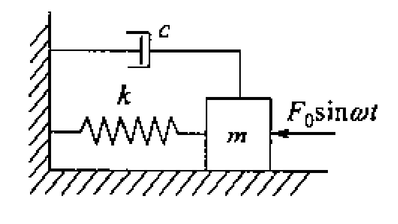


图 12.2.4-1

【例 12.2.4-1】 图 12.2.4-1 是谐迫振动系统。已知 $k=43.8$(N/cm)，$m=18.2$(kg)，$c=1.49$(N·s/cm)，$F_0=44.5$(N)，$\omega=15$(rad/s)。求系统的响应。

运行 vtb2(18.2,1.49,43.8,1,1,100,15,44.5)，可得出振动响应(图 12.2.4-2)。

读者可自己调整 c,ω 的大小，从而调整 ξ,λ 的大小，分析系统的响应形态。

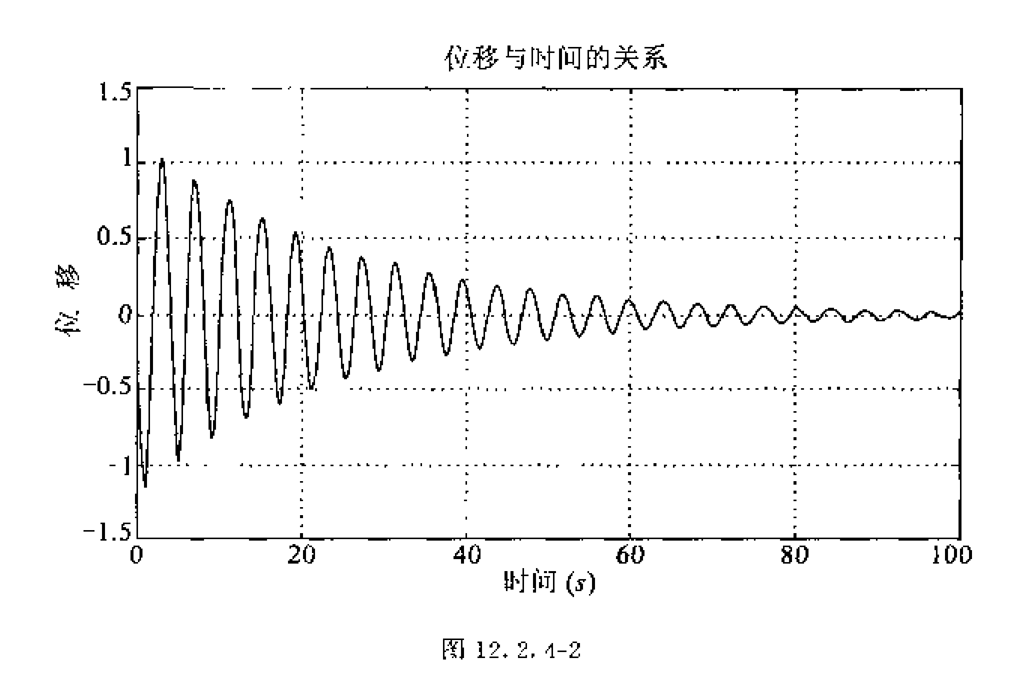


图 12.2.4-2

12.3　机械振动的仿真

在上一节，我们通过求解振动微分方程的通解的方法，计算了振动系统的响应。这种方法可以求解单自由度系统、非周期性激励、非线性振动。下面采用计算机仿真方法，求解运动微分方程的数值解，以研究振动系统在特定条件下的振动特性。

下面介绍的方法，既适用于单自由度，又能用于多自由度系统。对于多自由度系统，各变量均用矩阵表示，$\boldsymbol{M}$、$\boldsymbol{C}$、$\boldsymbol{K}$ 为方阵，$\boldsymbol{x}(t)$、$\dot{\boldsymbol{x}}(t)$、$\ddot{\boldsymbol{x}}(t)$、$\boldsymbol{f}(t)$ 均为列阵。

12.3.1　欧拉法及其改进

对于式(12.2.4-1)，为了求解振动响应，首先计算 Δt 秒后的状态，其次计算下一个 Δt 秒后的状态，这样逐步计算下去，则有

$$\left.\begin{aligned} x(t+\Delta t) &= x(t)+\dot{x}(t)\Delta t \\ \dot{x}(t+\Delta t) &= \dot{x}(t)+\ddot{x}(t)\Delta t \\ \ddot{x} &= \{f(t)-kx-c\dot{x}\}/m \end{aligned}\right\} \tag{12.3.1-1}$$

这种方法即为欧拉法。

若 Δt 足够小，欧拉法可以得到良好的结果。实际上，理论分析表明，$\Delta t\to 0$ 时，

欧拉法算出的数值收敛于精确解。

对于产生的误差可以通过 Taylor 级数展开加以分析。

由于 $t \to t+\Delta t$ 之间速度 $\dot{x}$ 的值是变化的，$t+\Delta t$ 时位移的精确解应为

$$x(t+\Delta t) = x(t) + \int_t^{t+\Delta t} \dot{x}(t)\mathrm{d}t \qquad (12.3.1\text{-}2)$$

对式(12.3.1-2)作积分变换

$$x(t+\Delta t) = x(t) + \frac{\Delta t}{1!}\dot{x}(t) + \int_t^{t+\Delta t}(t+\Delta t-\tau)\ddot{x}(\tau)\mathrm{d}\tau \quad (12.3.1\text{-}3)$$

最后可得到精确解

$$x(t+\Delta t) = x(t) + \frac{\Delta t}{1!}\dot{x}(t) + \frac{(\Delta t)^2}{2!}\ddot{x}(t) + \frac{(\Delta t)^3}{3!}\dddot{x}(t) + \cdots = \sum_{k=0}^{\infty}\frac{(\Delta t)^k}{k!}x^{(k)}(t)$$

(12.3.1-4)

由此可得：

(1)欧拉法是取 Taylor 级数展开式的前两项的解法。

(2)每前进一时间步 Δt，引起的误差为

$$R = \int_t^{t+\Delta t}(t+\Delta t-\tau)\ddot{x}(\tau)\mathrm{d}\tau = \frac{(\Delta t)^2}{2}\ddot{x}(\xi)$$

其中 $\ddot{x}(\xi)$ 是 $t \to t+\Delta t$ 时间的平均值，R 是由精确值减去计算值而得，这意味着 $R>0$ 时，计算值过小，反之亦然。

(3)在振动问题中应用欧拉法时，在波峰部分 $\ddot{x}<0$，因而 $R<0$，即计算值过大；在波谷部分 $\ddot{x}>0$，即计算值过小。总之，与精确解相比，数值解的振幅有逐渐增大的趋势。

下面是图 12.2.4-1 所示谐迫振动仿真的程序。

```
function vtb2(m,c,k,x0,v0,tf,w,f0)
%单自由度系统的谐迫振动
wn=sqrt(k/m);
z=c/2/m/wn; %阻尼比
lan=w/wn; %频率比
wd=wn*sqrt(1-z^2);
A=sqrt(((v0+z*wn*x0)^2+(x0*wd)^2)/wd^2);
delt=tf/1000;
t=0:delt:tf;
phi=atan2(2*z*lam,1-lam^2) %相位角
B=wn^2*f0/k/sqrt((wn^2-w^2)^2+(2*z*wn*w)^2);
x=A*exp(-z*wn*t).*sin(sqrt(1-z^2)*wn*t+phi)+B*sin(w*t+phi);
plot(t,x),grid
```

```
xlabel('时间(s)')
ylabel('位移')
title('位移与时间的关系')
```

请读者运行 vtb3(18.2,1.49,43.8,1,1,100,15,44.5,1)，并改变时间步长 delt 的值为 0.1,0.01,0.001，结果与图 12.2.4-1 相比较。

为了改进欧拉法的计算精度，采取以下两个基本方针：

(1) 对 Taylor 级数取更高次项。

(2) 对式(12.3.1-1)的积取具有更高精度的近似式。

若采取基本方针(1)，则

$$\left.\begin{aligned}\ddot{x}(t) &= (f(t) - kx(t) - c\dot{x}(t))/m \\ \dddot{x}(t) &= (\dot{f}(t) - k\dot{x}(t) - c\ddot{x}(t))/m \\ \dot{x}(t+\Delta t) &= \dot{x}(t) + \ddot{x}(t)\Delta t + \dddot{x}(t)(\Delta t)^2/2 \\ x(t+\Delta t) &= x(t) + \dot{x}(t)\Delta t + \ddot{x}(t)(\Delta t)^2/2\end{aligned}\right\} \qquad (12.3.1\text{-}5)$$

运用改进后的欧拉法编写的程序如下：

```
function vtb4(m,c,k,x0,v0,tf,w,f0,delt)
%用改进的欧拉法计算单自由度系统谐迫振动响应
wn=sqrt(k/m);%计算固有频率
fid1=fopen('disp','wt');   %建立一个位移文件 dip.dat
for t=0:delt:tf; %delt 为时间步长
    xdd=(f0*sin(w*t)-k*x0-c*v0)/m;   %计算加速度
    x3d=(f0*w*cos(w*t)-k*v0-c*xdd)/m;
    xd=v0+dd*delt+x3d*delt^2/2;   %计算速度
    x=x0+xd*delt+xdd*delt^2/2;   %计算位移
    fprintf(fid1,'%10.4f',x0);   %向文件中写数据
    x0=x;v0=xd;
    t
end
fid2=fopen('disp','rt');   %打开 disp.dat 文件
n=tf/delt;   %disp.dat 文件中位移的个数
x=fscanf(fid2,'%f',[1,n]);%将 disp.dat 文件中位移写成矩阵
t=1:n;
plot(t,x),grid
xlabel('时间(s)')
ylabel('位移')
title('位移与时间的关系')
```

若采取方针(2)，用数值积分的梯形公式，令

$$x(t+\Delta t)=x(t)+\frac{\dot{x}(t)+\dot{x}(t+\Delta t)}{2}\Delta t \qquad (12.3.1\text{-}6)$$

这个方法称梯形法。为了提高精度，还可考虑其他方法。若在区间$[t,t+\Delta t]$采用辛普生(Simpson)公式，则

$$x(t+\Delta t)=x(t)+\frac{\dot{x}(t)+4\dot{x}(t+\Delta t/2)+\dot{x}(t+\Delta t)}{6}\Delta t \qquad (12.3.1\text{-}7)$$

12.3.2　线性加速度法

振动问题的常用解法还有线性加速度法。它是综合上述两个方针得出的方法。基本公式为

$$x(t+\Delta t)=x(t)+\Delta t\dot{x}(t)+\frac{(\Delta t)^2}{2}\ddot{x}(t)+\frac{(\Delta t)^2}{6}\ddot{x}(t+\Delta t) \qquad (12.3.2\text{-}1)$$

$$\dot{x}(t+\Delta t)=\dot{x}(t)+\Delta t\frac{\ddot{x}(t)+\ddot{x}(t+\Delta t)}{2} \qquad (12.3.2\text{-}2)$$

式(12.3.2-2)是对变量$\dot{x}(t)$运用梯形法[式(12.3.1-6)]得出的式子。

其次改写式(12.3.2-1)为

$$x(t+\Delta t)=x(t)+\Delta t\dot{x}(t)+\frac{(\Delta t)^2}{2!}\ddot{x}(t)+\frac{(\Delta t)^3}{3!}\frac{\ddot{x}(t+\Delta t)-\ddot{x}(t)}{\Delta t} \qquad (12.3.2\text{-}3)$$

此式大致相当于取到 Taylor 展开式的三次项。它的物理意义是假定从时刻t到时刻$t+\Delta t$的加速度成直线变化。

线性加速度法与欧拉法不同，它属于隐式解法类型，计算时要多费点功夫。式(12.3.2-1)、式(12.3.2-2)是当仿真运行到t时计算$t+\Delta t$状态的公式。这时$x(t)$，$\dot{x}(t)$，$\ddot{x}(t)$是已知的，但$x(t+\Delta t)$，$\dot{x}(t+\Delta t)$，$\ddot{x}(t+\Delta t)$是未知的，三个未知数只有两个方程，故应联合运动微分方程式求解。由于问题的性质不同，联合的运动微分方程也不同。例如对于

$$m\ddot{x}(t)+c\dot{x}(t)+kx(t)=f(t)$$

式(12.3.2-1)、式(12.3.2-2)可联合式(12.3.2-4)

$$m\ddot{x}(t+\Delta t)+c\dot{x}(t+\Delta t)+kx(t+\Delta t)=f(t+\Delta t) \qquad (12.3.2\text{-}4)$$

所谓隐式法是不断求出新的时刻满足微分方程式的近似解。虽然计算工作量较大，但因精度和稳定性都较好，所以常被采用。另一种解法是显式解法。

解联立方程大致有直接法(代入法、消去法)和迭代法两种。这里介绍直接解法。

直接解法一：

将式(12.3.2-1)、(12.3.2-2)代入运动微分方程式(12.3.2-4)中

$$m\ddot{x}(t+\Delta t)+c\{\dot{u}(t)+\frac{\ddot{u}(t)+\ddot{u}(t+\Delta t)}{2}\Delta t\}+$$

$$k\{x(t)+\Delta t\dot{x}(t)+\frac{(\Delta t)^2}{3}\ddot{x}(t)+\frac{(\Delta t)^2}{6}\ddot{x}(t+\Delta t)\}=f(t+\Delta t)$$

$$\ddot{x}(t+\Delta t)=\frac{f(t+\Delta t)-c\{\dot{x}(t)+\frac{\Delta t}{2}\ddot{x}(t)\}-k\{x(t)+\Delta t\dot{x}(t)+\frac{(\Delta t)^2}{3}\ddot{x}(t)\}}{m+(\Delta t/2)c+\{(\Delta t)^2/6\}k}$$

(12.3.2-5)

上式右边均为已知量，将上式代回式(12.3.2-1)、式(12.3.2-2)可求 $x(t+\Delta t)$，$\dot{x}(t+\Delta t)$。

对于多自由度系统

$$MX+C\dot{X}+KX=F$$

则

$$\left.\begin{aligned}\ddot{X}(t+\Delta t)&=[M+\frac{\Delta t}{2}C+\frac{(\Delta t)^2}{6}K]^{-1}\{F(t+\Delta t)-C[\dot{X}(t)+\frac{\Delta t}{2}\ddot{X}(t)]-\\&\qquad K[X(t)+\Delta t\dot{X}(t)+\frac{(\Delta t)^2}{3}\ddot{X}(t)]\}\\\dot{X}(t+\Delta t)&=\dot{X}(t)+\Delta t\frac{\ddot{X}(t)+\ddot{X}(t+\Delta t)}{2}\\X(t+\Delta t)&=X(t)+\Delta t\dot{X}(t)+\frac{(\Delta t)^2}{3}\ddot{X}(t)+\frac{(\Delta t)^2}{6}\ddot{X}(t+\Delta t)\end{aligned}\right\}\quad(12.3.2\text{-}6)$$

因为式中出现的矩阵 $M+(\Delta t/2)C+\{(\Delta t)^2/6\}K$ 在各阶段都相同，可在第一次计算中预先求出逆矩阵。

直接解法二：

在直接解法一中，先消去 $x(t+\Delta t)$ 和 $\dot{x}(t+\Delta t)$，但也可以先消去 $\dot{x}(t+\Delta t)$ 和 $\ddot{x}(t+\Delta t)$ 求 $x(t+\Delta t)$。为此，由式(12.3.1-1)求 $\ddot{x}(t+\Delta t)$，将其代入式(12.3.2-2)，求 $\dot{x}(t+\Delta t)$，最后代入式(12.3.2-4)求 $x(t+\Delta t)$，即

$$x(t+\Delta t)=\frac{m\{2\ddot{x}(t)+\frac{6}{\Delta t}\dot{x}(t)+\frac{6}{(\Delta t)^2}x(t)\}+c\{\frac{\Delta t}{2}\ddot{x}(t)+2\dot{x}(t)+\frac{3}{\Delta t}x(t)\}+f(t+\Delta t)}{k+3c/\Delta t+6m/(\Delta t)^2}$$

(12.3.2-7)

对于多自由度系统，则

$$\left.\begin{aligned}
X(t+\Delta t)&=\{K+\frac{3}{\Delta t}C+\frac{6}{(\Delta t)^2}M\}^{-1}[M(2\ddot{X}(t)+\frac{6}{\Delta t}\dot{X}(t)+\frac{6}{(\Delta t)^2}X(t))\\
&+C\{\frac{\Delta t}{2}\ddot{X}(t)+2\dot{X}(t)+\frac{3}{\Delta t}X(t)\}+F(t+\Delta t)]\\
\dot{X}(t+\Delta t)&=\frac{3}{\Delta t}\{X(t+\Delta t)-X(t)\}-2\dot{X}(t)-\frac{\Delta t}{2}\ddot{X}(t)\\
\ddot{X}(t+\Delta t)&=\frac{6}{(\Delta t)^2}\{X(t+\Delta t)-X(t)\}-\frac{6}{\Delta t}\dot{X}(t)-2\ddot{X}(t)
\end{aligned}\right\}\quad(12.3.2\text{-}8)$$

式(12.3.2-8)是由威尔逊提出，并得到广泛的采用。式中出现的$1/\Delta t$，$1/(\Delta t)^2$等项并不是为了减少误差而采用的特殊方法所引入的，而是原封不动地保留消去过程中所出现的各项而已。将上式如下改写更为自然。

$$\left.\begin{aligned}
X(t+\Delta t)&=\{M+\frac{\Delta t}{2}C+\frac{(\Delta t)^2}{3!}K\}^{-1}[M\{\ddot{X}(t)+\Delta t\,\dot{X}(t)+\frac{(\Delta t)^2}{3}\ddot{X}(t)\}\\
&+C\{\frac{\Delta t}{2}\ddot{X}(t)+\frac{(\Delta t)^2}{3}\dot{X}(t)+\frac{(\Delta t)^3}{12}\ddot{X}(t)\}+\frac{(\Delta t)^2}{6}F(t+\Delta t)]\\
\ddot{X}(t+\Delta t)&=\frac{3!}{(\Delta t)^2}[X(t+\Delta t)-\{X(t)+\Delta t\,\dot{X}(t)+\frac{(\Delta t)^2}{3}\ddot{X}(t)\}]\\
\dot{X}(t+\Delta t)&=\dot{X}(t)+\Delta t\,\frac{\ddot{X}(t)+\ddot{X}(t+\Delta t)}{2}
\end{aligned}\right\}\quad(12.3.2\text{-}9)$$

【例 12.3.2-1】 如图 12.3.2-1 所示的三自由度弹簧-质量系统。试用直接解法二求系统的振动响应。

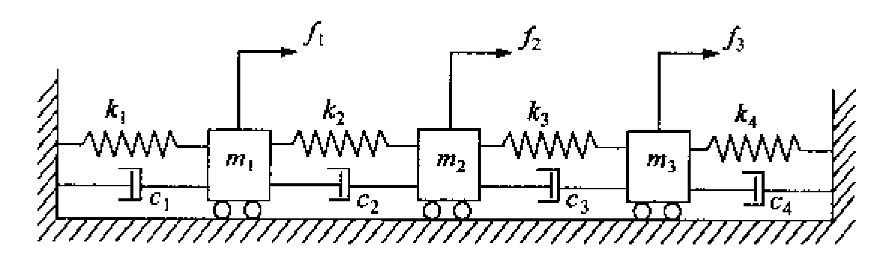


图 12.3.2-1

解：设$m_1=m_2=m_3=m_4=2(\text{kg})$，$c_1=c_2=c_3=c_4=1.5(\text{N}\cdot\text{s/m})$，$k_1=k_2=k_3=k_4=50(\text{N/m})$，$f_1=2.0\sin 3.754\text{t}$，$f_2=-2.0\cos 2.2\text{t}$，$f_3=1.0\sin 2.8\text{t}$。

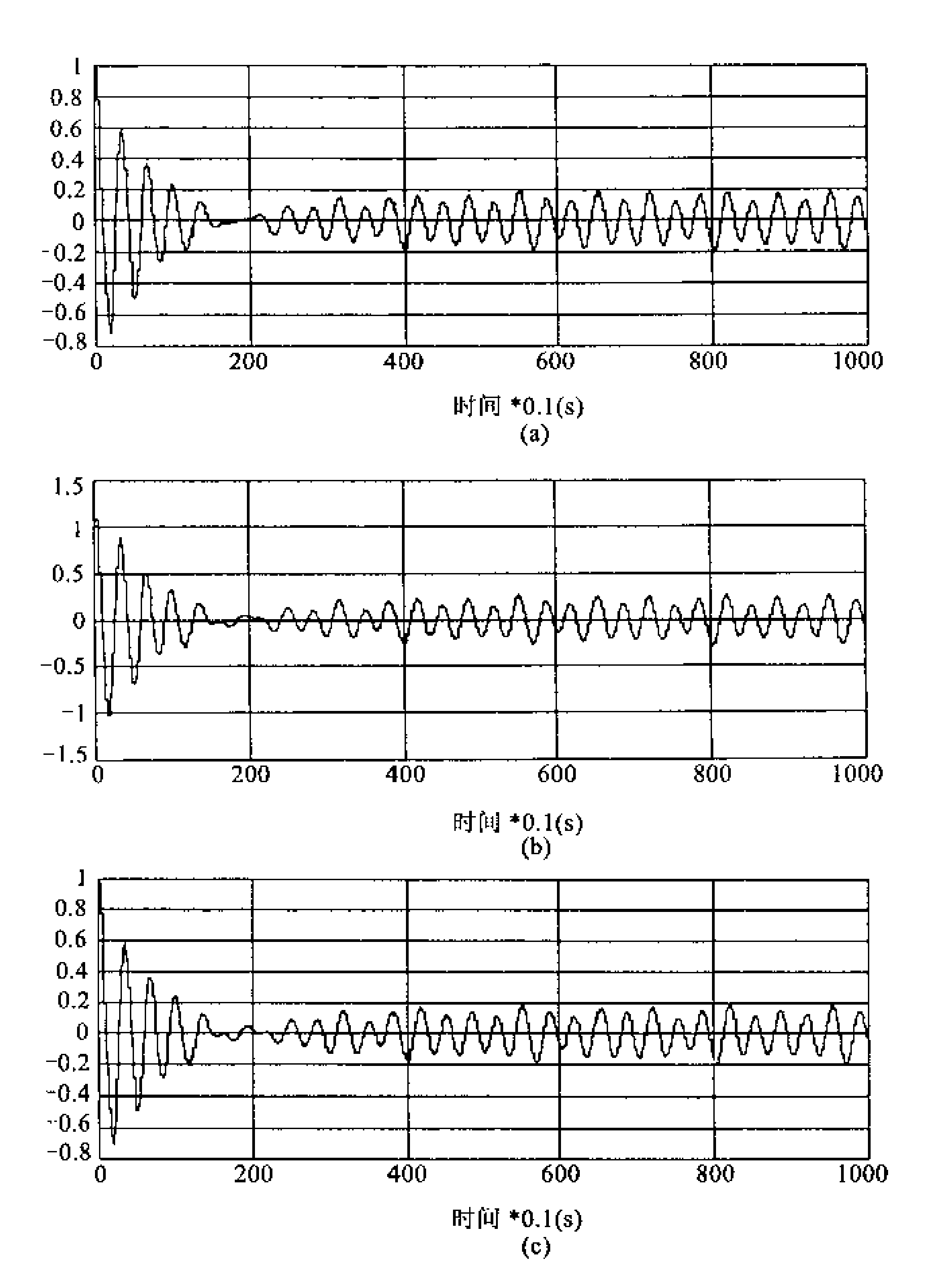
首先建立运动微分方程$\boldsymbol{M}\ddot{x}+\boldsymbol{C}\dot{x}+\boldsymbol{K}x=\boldsymbol{F}$，其中

$$\boldsymbol{M}=2\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix},\boldsymbol{C}=1.5\begin{bmatrix}2&-1&0\\-1&2&-1\\0&-1&2\end{bmatrix},\boldsymbol{K}=50\begin{bmatrix}2&-1&0\\-1&2&-1\\0&-1&2\end{bmatrix},$$

$\boldsymbol{F}=[2.0\sin 3.754t,-2.0\cos 2.2t,1.0\sin 2.8t]$。

由式(12.3.2-6)编写的计算程序如下：

```
function vtb5(tf,delt)
%用线性加速度法计算三自由度系统谐迫振动响应
close all;clc
fid1 =fopen('disp','wt');  %建立一个位移文件dip.dat
m=2*[1 0 0;0 1 0;0 0 1];  %质量矩阵
c=1.5*[2 -1 0;-1 2 -1;0 -1 2];  %阻尼矩阵
k=50*[2 -1 0; -1 2 -1;0 -1 2];  %刚度矩阵
x0=[1 1 1]';  %初始位移
v0=[1 1 1]';  %初始速度
bita =1/6;
md=inv(m+delt/2*c+bita*delt^2*k);
m1=inv(m);
[E,F]=eig(m1*k);
diag(sqrt(F));  %计算固有频率(显示)
for t=0:delt:tf;  %delt 为时间步长
    f=[2.00*sin(3.754*t) -2.00*cos(2.2*t) 1.00*sin(2.8*t)]';
    if t=0: xdd0=m1*(f-k*x0-c*v0);  %计算初始加速度
    else
    xdd =md*(f-c*(v0+delt/2*xdd0)...
           -k*(x0+delt*v0+(1/2-bita)*delt^2*xdd0));  %计算加速度
x=md*(m*(x0+delt*v0+delt^2/3*xdd0)+c*(delt/2*x0+delt^2/3*v0+
delt^3/12*xdd0)+delt^2/6*f);  %计算位移
       xd=v0+delt/2*(xdd0+xdd);  %计算速度
       xdd0=xdd;v0=xd;x0=x;
       fprintf(fid1,'%10.4f',x0);  %向文件中写数据
       t  %显示计算时间步长
end
fid2=fopen('disp','rt');  %打开disp.dat 文件
n=tf/delt;  %disp.dat 文件中位移的个数
x=fscanf(fid2,'%f',[3,n]);  %将disp.dat 文件中位移写成矩阵
t=1:n;
    figure('numbertitle','off','name','自由度1的位移','pos',[450 180 400 420]);
    plot(t,x(1,t)),grid,xlabel('时间*0.1秒'),title('自由度1的位移与时间的关
系')
    figure('numbertitle','off','name','自由度2的位移','pos',[350 160 400 420]);
    plot(t,x(2,t)),grid,xlabel('时间*0.1秒'),title('自由度2的位移与时间的关
```



(a)自由度1的位移与时间的关系;(b)自由度－2的位移与时间的关系;(c)自由度－3的位移与时间的关系

图12.3.2 2

```
系')
    figure('numbertitle','off','name','自由度3的位移','pos',[250 140 400 420]);
    plot(t,x(3,t)),grid,xlabel('时间*0.1秒'),title('自由度3的位移与时间的关
系')
```

运行 VTB5(100,0.1),则可显示三个自由度的位移随时间的变化关系,如图 12.3.2-2 所示。

12.3.3 纽马克-β法

纽马克-β法是线性加速度法方法之一,它是将式(12.3.2-3)最后一项的系数 1/3! 改为参数 β,通常可写成

$$X(t+\Delta t)=X(t)+\Delta t\,\dot{X}(t)+\{(\Delta t)^2/2\}\ddot{X}(t)+\beta(\Delta t)^2\{\ddot{X}(t+\Delta t)-\ddot{X}(t)\} \tag{12.3.3-1}$$

$$\dot{X}(t+\Delta t)=\dot{X}(t)+\Delta t[\ddot{X}(t)+\ddot{X}(t+\Delta t)]/2 \tag{12.3.3-2}$$

β 是调节公式特性的参数,一般取值范围为 0≤β≤1/2,实际上往往固定采用 β=1/6,因此在多数情况下纽马克法是线性加速度法的别名。此外也常采用 β=1/4。计算方法与线性加速度法一样。若采用前面介绍的"直接解法一"的方法,计算式如下:

$$\ddot{X}(t+\Delta t)=\left\{M+\frac{\Delta t}{2}C+\beta(\Delta t)^2K\right\}^{-1}\Big[F(t+\Delta t)-C[\dot{X}(t)+\frac{\Delta t}{2}\ddot{X}(t)] \\ -K\Big[X(t)+\Delta t\,\dot{X}(t)+(\frac{1}{2}-\beta)(\Delta t)^2\ddot{X}(t)\Big] \tag{12.3.3-3}$$

$$\dot{X}(t+\Delta t)=\dot{X}+\frac{\Delta t}{2}[\ddot{X}(t)+\ddot{X}(t+\Delta t)]$$

$$X(t+\Delta t)=X(t)+\frac{\Delta t}{1!}\dot{X}(t)+\frac{(\Delta t)^2}{2!}\ddot{X}(t)+\beta(\Delta t)^3\frac{\ddot{X}(t+\Delta t)-\ddot{X}(t)}{\Delta t}$$

```
function vtb6(tf,delt)
%用纽马克法计算三自由度系统谐迫振动响应
close all; clc
fid1=fopen('disp','wt');   %建立一个位移文件 dip.dat
m=2*[1 0 0;0 1 0;0 0 1];   %质量矩阵
c=1.5*[2 -1 0;-1 2 -1;0 -1 2];   %阻尼矩阵
k=50*[2 -1 0;-1 2 -1;0 -1 2];   %刚度矩阵
x0=[1 1 1]';   %初始位移
v0=[1 1 1]';   %初始速度
bita=1/6;
md=inv(m+delt/2*c+bita*delt^2*k);
```

```
m1=inv(m);
[E,F]=eig(m1*k);
diag(sqrt(F));  %计算固有频率(显示)
for t=0:delt:tf; %delt 为时间步长
    f=[2.00*sin(3.754*t)  -2.00*cos(2.2*t) 1.00*sin(2.8*t)]';
    if t==0; xdd0=m1*(f-k*x0-c*v0);  %计算初始加速度
    else
            xdd=md*(f-c*(v0+delt/2*xdd0)...
              -k*(x0+delt*v0+(1/2-bita)*delt^2*xdd0));  %计算加速度
            xd=v0+delt/2*(xdd0+xdd);  %计算速度
            x=x0+delt*v0+delt^2/2*xdd0+bita*delt^3*(xdd-xdd0)/delt;
      %计算位移
    v0=xd;x0=x;
    fprintf(fid1,'%10.4f',x0);  %向文件中写数据
    t  %显示计算时间步长
end
fid2=fopen('disp','rt');  %打开 disp.dat 文件
n=tf/delt;  %disp.dat 文件中位移的个数
x=fscanf(fid2,'%f',[3,n]);  %将 disp.dat 文件中位移写成矩阵
t=1:n;
figure('numbertitle','off','name','自由度 1 的位移','pos',[450 180 400 420]);
plot(t,x(1,t)),grid,xlabel('时间*0.1 秒'),title('自由度 1 的位移与时间的关系')
figure('numbertitle','off','name','自由度 2 的位移','pos',[350 160 400 420]);
plot(t,x(2,t)),grid,xlabel('时间*0.1 秒'),title('自由度 2 的位移与时间的关系')
figure('numbertitle','off','name','自由度 3 的位移','pos',[250 140 400 420]);
plot(t,x(3,t)),grid,xlabel('时间*0.1 秒'),title('自由度 3 与时间的关系')
```

12.3.4 威尔逊-θ 法

这个方法也是线性加速度法的变形。它的特点是把线性加速度法进一步扩展。计算步骤与线性加速度法大致相同，所不同的是线性加速度法在时刻$(t+\Delta t)$使用运动方程，而威尔逊-θ 法则应用于更后一点的时刻$(t+\theta\Delta t)(\theta>1)$，即

$$\left.\begin{aligned}X(t+\theta\Delta t)&=X(t)+\theta\Delta t\,\dot{X}(t)+\frac{(\theta\Delta t)^2}{3}X(t)+\frac{(\theta\Delta t)^2}{6}\ddot{X}(t+\theta\Delta t)\\ \dot{X}(t+\theta\Delta t)&=\dot{X}(t)+\theta\Delta t\,\frac{X(t)+X(t+\theta\Delta t)}{2}\end{aligned}\right\}\quad(12.3.4\text{-}1)$$

与下式联立求解

$$M X(t+\theta\Delta t)+C\dot{X}(t+\theta\Delta t)+KX(t+\theta\Delta t)=F(t+\theta\Delta t)$$

对求得的 $X(t+\theta\Delta t)$ 内插求 $\dot{X}(t+\Delta t)$：

$$\ddot{X}(t+\Delta t)=\frac{(\theta-1)\dot{X}(t)+\dot{X}(t+\theta\Delta t)}{\theta} \tag{12.3.4-2}$$

再将结果代入下列两式求 $X(t+\Delta t)$，$\dot{X}(t+\Delta t)$：

$$X(t+\Delta t)=X(t)+\Delta t\,\dot{X}(t)+(\Delta t)^2 X(t)/3+(\Delta t)^2 X(t+\Delta t)/6 \tag{12.3.4-3}$$

$$\dot{X}(t+\Delta t)=\dot{X}(t)+\Delta t\{X(t)+\ddot{X}(t+\Delta t)\}/2 \tag{12.3.4-4}$$

本方法的物理意义是：加速度在时刻 t 到 $t+\theta\Delta t$ 内为线性变化，首先计算[t，$t+0\Delta t$]区间近似解，但仅取其中前半部分（到时刻 $t+\Delta t$ 为止）作为近似解而舍去后半部分（时间 $t+\Delta t$ 以后）。这种巧妙的处理方法并非出于物理的原因，而主要是数学的理由。要理解这一点首先应了解数值计算的稳定性的知识，这里作一简单介绍。

振动仿真的失败原因之一往往是步长幅度过大。程序是正确的，输入信息也没有错误，但却得到异常的结果。仔细研究一下可发现：计算刚开始时结果比较正常，但在计算过程中出现异常现象，绝对值迅速增大以致几乎溢出。这种症状称为不稳定现象。产生这种现象的原因很多，也不一定只是取值方面的问题，但通常即使是良态方程，若 Δt 过大，多半还是会出现不稳定现象。

那么 Δt 究竟取多大才安全呢？虽然 Δt 的安全值可按公式计算，但通常取小于周期的 1/6。例如对于单自由度系统，当频率为 10Hz，周期为 0.1s 时，取 Δt 小于 1/60s就稳定。对多自由度系统，则 Δt 应小于最短周期的 1/6，例如若周期为10s，0.1s和 0.01s 三种波形混合的混合载波系统，则必须取 Δt 小于 1/600s，计算才稳定。

但只是在需要详细研究小波时才这样做，一般在振动计算中起重要作用的是大波，详细研究小波意义不大。实际上，导致结构物破坏的原因主要是低阶振型，而高阶振型的影响是局部的。后者振幅不大，衰减也快，影响不大。为了次要因素而把 Δt 取得过小是不经济的。

特别是用有限元法分析振动时，若单元分得很细而考虑到第数百阶的高阶振型的话，则为了与此相称必须把 Δt 取得很小，这就要耗费相当长的计算时间。

在威尔逊-θ 法中，只要取 θ 大于 1.37 以上，不管 Δt 取怎样的值都是稳定的，即这种算法是无条件稳定的。当然，Δt 过大，精度要降低，但只要不发散，就可根据经验和工程常识判断，灵活掌握。例如，Δt 取结构基本周期（最长的周期）的 1%左右即可得到相当满意的结果。

因此，威尔逊-θ 法是实用价值很高的出色的解法，虽然由于增加了参数 θ，看

起来式子稍复杂一些，但计算工作量与线性加速度法和纽马克-β 法差不多。

θ 取值小于1.37意义不大。但并不是说取值只要在1.37以上，不管多大都可以。实际上，θ 最好不要太大，否则精度下降。例如 $\Delta t=0.01$ 秒，$\theta=3\times10^{10}$，就意味着假定此后十年内加速度均为线性变化，且据十年后的加速度来计算0.01秒后的状态，这样的不合理情况当然精度是很差的，即使 $\theta=2$，误差已相当显著。因此，威尔逊-θ 法的合理的 θ 值为1.4。

威尔逊-θ 法通常取 Δt 较大，用直接法为好。

按"直接法一"的计算公式：

$$\left.\begin{aligned}
\ddot X(t+\theta\Delta t)=&\{M+\frac{\theta\Delta t}{2}C+\frac{(\theta\Delta t)^2}{6}K\}^{-1}[F(t+\theta\Delta t)-C\{\dot X(t)+\frac{\theta\Delta t}{2}\ddot X(t)\}\\
&-K\{X(t)+\theta\Delta t\,\dot X(t)+\frac{(\theta\Delta t)^2}{3}\ddot X(t)\}]\\
\ddot X(t+\Delta t)=&\{(\theta-1)\ddot X(t)+\ddot X(t+\theta\Delta t)\}/\theta\\
\dot X(t+\Delta t)=&\dot X(t)+\Delta t\{\ddot X(t)+\ddot X(t+\Delta t)\}/2\\
X(t+\Delta t)=&X(t)+\Delta t\,\dot X(t)+(\Delta t)^2\ddot X(t)/3+(\Delta t)^2\ddot X(t+\Delta t)/3
\end{aligned}\right\}\tag{12.3.4-5}$$

按"直接法二"的计算公式：

$$\left.\begin{aligned}
X(t+\theta\Delta t)=&\{K+\frac{3C}{\theta\Delta t}+\frac{6}{(\theta\Delta t)^2}M\}^{-1}[M\{2\ddot X(t)+\frac{6}{\theta\Delta t}\dot X(t)+\frac{6}{(\theta\Delta t)^2}X(t)\}\\
&+C\{\frac{\theta\Delta t}{2}\ddot X(t)+2\dot X(t)+\frac{3}{\theta\Delta t}X(t)\}+F(t+\Delta t)]\\
\ddot X(t+\theta\Delta t)=&\frac{6}{(\theta\Delta t)^2}\{X(t+\theta\Delta t)-X(t)\}-\frac{6}{\theta\Delta t}\dot X(t)-2\ddot X(t)\\
\ddot X(t+\Delta t)=&(1-\frac{1}{\theta})\ddot X(t)+\frac{1}{\theta}\ddot X(t+\theta\Delta t)\\
\dot X(t+\Delta t)=&\dot X(t)+(\Delta t/2)\{\ddot X(t)+\ddot X(t+\Delta t)\}\\
X(t+\Delta t)=&X(t)+\Delta t\,\dot X(t)+(\Delta t)^2\{2\ddot X(t)+\ddot X(t+\Delta t)\}/6
\end{aligned}\right\}\tag{12.3.4-6}$$

通常流行的是后一组公式。

图12.3.2-1所示的三自由度振动系统，按威尔逊-θ 法计算响应的编程如下。

```
function vtb7(tf,delt)
%用威尔逊-θ法计算三自由度系统谐迫振动响应
close all;clc
fid1=fopen('disp','wt');  %建立一个位移文件dip.dat
m=2*[1 0 0;0 1 0;0 0 1];  %质量矩阵
```

```
c  1.5*[2 -1 0;-1 2 -1;0 -1 2];  %阻尼矩阵
k=50*[2 -1 0;-1 2 -1;0 -1 2];  %刚度矩阵
x0=[1 1 1]';  %初始位移
v0=[1 1 1]';  %初始速度
theta=1.4;
md=inv(k+3*c/theta/delt+6/(theta*delt)^2*m);
m1=inv(m);
[E,F]=eig(m1*k);
diag(sqrt(F));  %计算固有频率(显示)
for t=0:delt:tf;  %delt 为时间步长
    f=[2.00*sin(3.754*t) -2.00*cos(2.2*t) 1.00*sin(2.8*t)]';
    if t=0; xdd0=m1*(f-k*x0-c*v0);  %计算初始加速度
    else
        xtheta=md*(m*(2*xdd0+6/theta/delt*v0+6/(theta*delt)^2*x0)
        +c*(theta*delt/2*xdd0+2*v0+3/theta/delt*x0)+f);
        %计算 u(t+θΔt)速度
        xddtheta=6/(theta*delt)^2*(xtheta-x0)-6/theta/delt*v0-2*xdd0;
      %计算加速度
        xdd=(1-1/theta)*xdd0+1/theta*xddtheta;  %计算加速度
        xd=v0+delt/2*(xdd0+xdd);  %计算速度
        x=x0+delt*v0+delt^2*(2*xdd0+xdd)/6;  %计算位移
        v0=xd;x0=x;xdd0=xdd;
        fprintf(fid1,'%10.4f',x0);  %向文件中写数据
        t  %显示计算时间步长
end
    fprintf(fid1,'%10.4f',x0);  %向文件中写数据
end
fid2=fopen('disp','rt');  %打开 disp.dat 文件
n=tf/delt;  %disp.dat 文件中位移的个数
x=fscanf(fid2,'%f',[3,n]);  %将 disp.dat 文件中位移写成矩阵
t=1:n;
figure('numbertitle','off','name','自由度 1 的位移','pos',[450 180 400 420]);
plot(t,x(1,t)),grid,xlabel('时间*0.1秒'),title('自由度 1 的位移与时间的关系')
figure('numbertitle','off','name','自由度 2 的位移','pos',[350 160 400 420]);
plot(t,x(2,t)),grid,xlabel('时间*0.1秒'),title('自由度 2 的位移与时间的关系')
figure('numbertitle','off','name','自由度 3 的位移','pos',[250 140 400 420]);
plot(t,x(3,t)),grid,xlabel('时间*0.1秒'),title('自由度 3 的位移与时间的关系')
```

习 题 十 二

1. 图1所示为一减振系统，已知 $k_1=8750(\text{N/m})$，$m=227(\text{kg})$，$C=350(\text{N}\cdot\text{s/m})$。系统开始静止，在给振动体一个冲击以后，它就开始以初速度 $V_0=0.127(\text{m/s})$ 沿 x 轴正向运动，试编程序求该系统的振动固有频率，及振动响应的位移、速度、加速度。（提示：系统中两弹簧为并联，构成动力学模型时总刚度 $k=2k_1$）

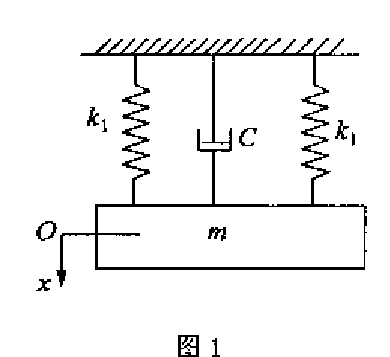


图1

2. 图2所示为二自由度振动系统。已知 $m_1=2(\text{kg})$，$m_2=m_1/2$，$k_1=k_2=500(\text{N/m})$，$C_2=250(\text{N}\cdot\text{s/m})$，$Q=2\sin70t$。用牛顿定律建立的运动微分方程为

$$m_1\ddot{x}_1+C(\dot{x}_1-\dot{x}_2)+(k_1+k_2)x_1-k_2x_2=Q$$
$$m_2\ddot{x}_2+C(\dot{x}_2-\dot{x}_1)+k_2(x_2-x_1)=0$$

开始时，系统静止，施加 Q 力后，系统以初速度 $V_{01}=0.2\text{m/s}$，$V_{02}=0$ 向 x 轴正向运动。试按纽马克-β 法计算系统的响应。

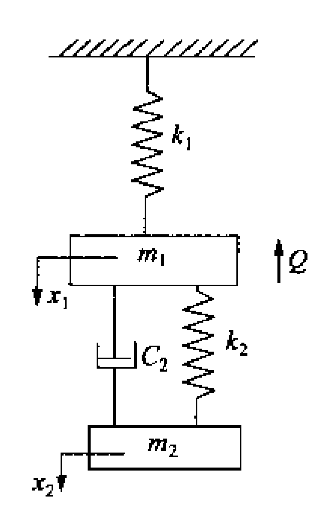


图2

3. 图3为三自由度无阻尼强迫振动系统。已知 $k_1=60(\text{N/m})$，$k_2=20(\text{N/m})$，$k_3=80(\text{N/m})$，$m_1=3(\text{kg})$，$m_2=1(\text{kg})$，$m_3=2(\text{kg})$，$F=1(\text{N})$。试求系统的固有频率，并用威尔逊-θ 法求系统的响应。

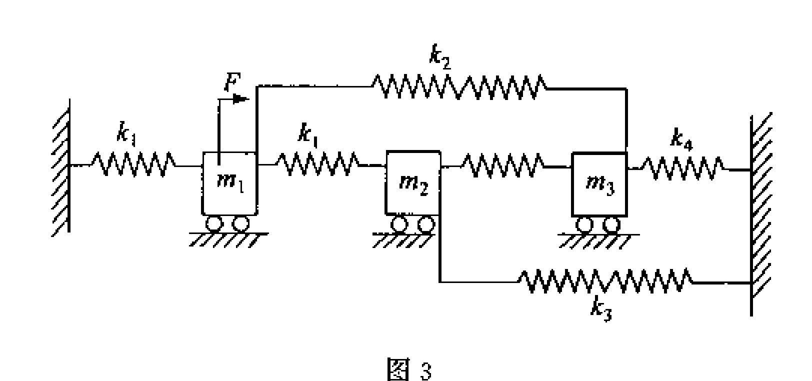


图3

附录一　SIMULINK 各模块的用途

在 SIMULINK 中，各个模块按照其不同的功能和作用分成了几个库，这些库包括：

1. Source library。这个库是由产生源信号和源数据的模块组成的。附录 1-1 列出了这个库的所有模块。

附录 1-1　Source library

模　块　名	用　　途
Band-Limited White Noise	把白噪声加到连续系统中
Chip Signal	产生一个频率不断增大的正弦波
Clock	显示和提供仿真时间
Constant	产生一个常值
Digital Clock	在规定的采样间隔产生仿真时间
Discrete Pulse Generator	在固定的时间间隔产生离散的脉冲
From File	从文件中读数据
From Workspace	从工作面上定义的矩阵中读数据
Pulse Generator	在固定的时间间隔产生脉冲
Ramp	产生稳定增加或减少的信号
Random Number	产生正态分布的随机数
Repeating Sequence	产生规律重复的任意信号
Signal Generator	产生各种不同的波形
Sine Wave	产生一个正弦波
Step	产生一个阶跃函数
Uniform Random Number	产生非均匀分布的随机数

2. Sinks library。这个库是由用于显示和输出的模块组成的。附录 1-2 列出了这个库中的所有模块。

附录 1-2 Sinks library

模 块 名	用 途
Auto-Scale Graph Scope	在Matlab自动调整显示比例的图形窗口显示信号
Display	显示输入的数值
Graph Scope	在Matlab图形窗口显示信号
Hit Crossing	在规定值附近增加仿真步数
Scope	在仿真过程中显示信号
Stop Simulation	当输入不为零时停止仿真
To File	把数据输入到文件中
To Workspace	把数据输出到工作面上定义的一个矩阵中
XY Graph Scope	在Matlab图形窗口显示信号的X-Y图

3. Discrete library。这个库是由描述离散时间系统的模块组成的。附录1-3列出了这个库中的所有模块。

附录 1-3 Discrete library

模 块 名	用 途
Discrete Filter	建立IIR和FIR过滤器
Discrete-Time Integrator	对一个信号进行离散积分
Discrete-Time Limited Integrator	对一个信号进行离散有限积分
Discrete State-Space	建立一个离散状态空间模型
Discrete Transfer Fcn	建立一个离散传递函数
Discrete Zero-Pole	以零极点形式建立一个离散传递函数
First-Order Hold	建立一阶采样保持器
Unit Delay	对一个信号延迟一个采样周期
Zero-Order Hold	建立一个采样周期的零阶保持器

4. Linear library。这个库由标准线性函数和线性系统的模块组成。附录1-4列出了这个库中的所有模块。

附录 1-4 Linear library

模 块 名	用 途
Derivative	对输入信号进行微分
Dot Product	对输入信号进行点积
Gain	对输入信号乘上一个常数增益
Integrator	对输入信号进行积分
Matrix Gain	对输入信号乘上一个矩阵增益
Slider Gain	以滑动形式改变增益
State-Space	建立一个线性状态空间模型
Sum	对输入信号进行求和
Transfer Fcn	建立一个线性传递函数
Zero-Pole	以零极点形式建立一个传递函数

5. Nonlinear library。这个库由描述非线性函数的模块组成。附录 1-5 列出了这个库中的所有模块。

附录 1-5 Nonlinear library

模 块 名	用 途
Abs	输出输入信号的绝对值
Algebraic Constraint	强迫输入信号为零
Backlash	用放映的方式模仿一个系统的特性
Combinatorial	建立一张真值表
Coulomb and Viscous Friction	在原点不连续而在原点以外具有线性增益
Dead Zone	提供一个死区
Fcn	对输入进行规定的表示
Hit Crossing	检查叉点
Limited Integrator	在规定的范围内进行积分
Logical Operator	对输入进行规定的逻辑运算
Lookup Table	对输入进行分段的线性映射

续表

模 块 名	用 途
Matlab Fcn	定义一个函数，对输入信号进行处理
Manual Switch	在两个输入之间进行切换的开关
Math Function	完成数学运算函数
Memory	输出本模块上一步的输入值
MinMax	输出最小或最大输入数值
Multiport Switch	选择块输入
Product	对输入进行乘积运算
Quantizer	对输入进行量化处理
Rate Limiter	限制信号的变化速率
Relational Operator	对输入进行一定的关系运算
Relay	在两个值中轮流输出
Reset Integrator	在仿真中对积分器进行重新初始化
Rounding Function	完成四舍五入取整运算函数
Saturation	对输入信号进行限幅
S-Function	S函数入口
Sign	符号函数
Transport Delay	对输入信号进行一定的延迟
2-D LookUp Table	对两个输入进行分段的线性映射
Variable Transport Delay	对输入信号进行不定量的延迟

6. Connections library。这个库是由一些实现模块与模块之间连接功能的模块组成的。附录1-6列出了这个库中的所有模块。

附录 1-6 Connections library

模块名	用途
Data Store Read	从数据存储器中读数据
Data Store Memory	定义一个数据存储器
Data Store Write	往数据存储器中写数据
Demux	把向量信号分开输出
Enable	为子系统增加一个入口
From	从 Goto 块接受输入
Goto Tag Visibility	定义 Goto 块标识范围
Ground	与一个未连接的输入口接地
IC	设置一个信号的初始值
Inport	给系统提供一个外部输入
Mux	把几个信号合并成向量形式
Outport	给系统提供一个外部输出
Subsystem	表示一个系统在另外一个系统中
Terminator	与一个未连接的输出口断开
Trigger	为子系统增加一个触发入口

附录二 MATLAB主要函数及命令表

附录2.1 系统管理基本命令

附录2.1.1 系统查询命令

demo	演示程序	type	显示文件内容
help	在线帮助命令	what	列出当前目录中的m文件、mat文件和mex文件
lookfor	关键词检索	which	确定指定函数和文件的位置
path	控制MATLAB的搜索路径		
pathtool	路径管理对话框		

附录2.1.2 变量和内存空间的管理

clear	从内存中清除变量	save	把内存变量存入磁盘
disp	显示矩阵和文字内容	size	确定矩阵的维数
length	确定向量的长度	who	显示工作内存中的变量名
load	从磁盘中调入数据变量	whos	显示工作内存中的变量细节
pack	合并工作内存中的碎块	workspace	工作内存浏览器

附录2.1.3 调用操作系统和文件处理

cd	改变当前工作目录	edit	矩阵编辑器
delete	删除文件	getenv	列出环境值
diary	储存MATLAB命令窗操作内容	!	执行外部应用程序
dir	将文件列表		

附录2.1.4 命令窗控制

clc	清除命令窗	echo	显示命令文件命令的切换开关
format	设置数据输出格式	more	命令窗口分页输出的控制开关
home	光标返回行首		

附录 2.1.5 MATLAB 的启动和终止

quit	退出 MATLAB	startup	启动 MATLAB 时自动执行 M 文件
matlabrc	MATLAB 的主启动文件		

附录 2.2 运算符和特殊算例

附录 2.2.1 逻辑操作

all	全非零元向量为真	isglobal	若是全局变量则为真
any	有非零元的向量为真	isinf	若是无穷大则为真
exist	检查变量或函数是否被定义	isnan	若为非数则为真
find	找出非零元素 1 的下标	isreal	若是实数矩阵则为真
finite	若是有限数则为真	issparse	若是稀疏矩阵则为真
isempty	若是空矩阵则为真	isstr	若是字符串则为真

附录 2.3 基本数学函数

附录 2.3.1 三角和超越函数

acos	反余弦	cos	余弦
acosh	反双曲余弦	cosh	双曲余弦
acot	反余切	cot	余切
acsc	反余割	coth	双曲余切
acsch	反双曲余割	csc	余割
asec	反正割	csch	双曲余割
asech	反双曲正割	sec	正割
asin	反正弦	sech	双曲正割
asinh	反双曲正弦	sin	正弦
atan	反正切	sinh	双曲正弦
stanh	反双曲正切	tan	正切
atan2	四象限反正切	tanh	双曲正切

附录 2.3.2 指数和对数函数

exp	指数	log10	常用对数
log	自然对数	sqrt	平方根

附录 2.3.3 复数函数

abs	绝对值	imag	复数虚部
angle	相角	real	复数实部
conj	复数共轭		

附录 2.3.4 数值处理

ceil	朝正无穷大方向取整	rem	求余数
fix	朝零方向取整	round	四舍五入取整
floor	朝负无穷大方向取整	sign	符号函数

附录 2.4 基本矩阵函数和操作

附录 2.4.1 基本矩阵

eye	单位阵	ones	全 1 矩阵
linspace	线性等分向量	rand	均匀分布随机阵
logspace	对数等分向量	randn	正态分布随机阵
meshgrid	三维曲面的分格坐标	zeros	全零矩阵

附录 2.4.2 特殊变量和常数

ans	最新表达式的运算结果	nargin	函数输入变量的个数
computer	计算机类型	nargout	函数输出变量的个数
eps	浮点相对误差	pi	3.1415926535897...
flops	浮点运算次数	realmax	最大正浮点数
i,j	虚数单位	realmin	最小正浮点数
inf	无穷大	version	MATLAB 版本
isieee	计算机采用 IEEE 算法则为真	why	一般问题的简明答案
NaN	非数		

附录 2.4.3 时间和日期

clock	时钟	etime	用 CLOCK 计算的时间
cputime	MATLAB 占用 CPU 时间	tic	计算时钟起动
date	日期	toc	计算时钟终止和显示

附录 2.4.4 矩阵操作

diag	创建对角阵，抽取对角向量	rot90	矩阵逆时针旋转 90 度
fliplr	矩阵的左右翻转	tril	抽取下三角阵
flipud	矩阵的上下翻转	triu	抽取上三角阵
reshape	矩阵变维	:	矩阵的援引和编辑

附录 2.5 字符串函数

附录 2.5.1 通用字符串函数

abs	把字符串变成 ASCII 码值	isstr	若是边际符串则为真
blanks	空格符号	setstr	把 ASCII 码值变为符号
deblank	删除最后的空格	strings	MATLAB 中的字符串
eval	执行串形式的 MATLAB	str2mat	把单个串变成文本矩阵

附录 2.5.2 串比较

isletter	串中是字母则为真	strcmp	比较字符串
isspace	串中是空格则为真	strrep	用一个串代替另一个串的子串
findstr	在一个串中寻找一个子串	strtok	删除串中的指定子串
lower	把字符串变成小写	upper	把字符串变成大写

附录 2.5.3 字符串与数值间的转换

int2str	把整数转换为串	sscanf	按格式把串转换为数字
num2str	把浮点数转换为串	str2num	把串转换为浮点数
sprintf	按格式把数字转换为串		

附录 2.5.4 十六进制数与十进制数间的转换

dec2hex	把 10 进制整数变成 16 进制串

hex2num	把16进制串变成IEEE浮点数
hex2dec	把16进制串变成10进制整数

附录2.6 矩阵函数和数值线性代数

附录2.6.1 矩阵分析

cond	矩阵条件数	rank	秩
det	行列式的值	rcond	LINPACK 逆条件数
norm	矩阵或向量范数	rref	转换为行阶梯形
null	零空间	trace	迹
orth	值空间		

附录2.6.2 线性方程

chol	Cholesky 分解	nnls	负最小二乘解
inv	矩阵的逆	pinv	伪逆
lscov	已知协方差的最小二乘解	qr	QR 分解
lu	LU 分解	\、/	解线性方程

附录2.6.3 特征值与奇异值

balance	改善特征值精度的平衡刻度	polyeig	多项式特征值问题
cdf2rdf	复数对角型转换到实块对角型	qz	广义特征值
eig	矩阵特征值和特征向量	rsf2csf	实块对角型转换到复数对角型
hess	Hessenberg 矩阵	schur	Schur 分解
ploy	特征多项式	svd	奇异值分解

附录2.6.4 矩阵函数

expm	矩阵指数	funm	计算一般矩阵函数
expm1	矩阵指数的 Pade 逼近	logm	矩阵对数
expm2	用泰勒级数求矩阵指数	sqrtm	矩阵平方根
expm3	通过特征值和特征向量求矩阵指数		

附录 2.7 数据分析和傅里叶变换

附录 2.7.1 基本运算

cumprod	元素累计积	prod	元素积
cumsum	元素累计和	sort	由小到大排序
max	最大值	std	标准差
mean	平均值	sum	元素和
median	中值	trapz	梯形数值积分
min	最小值		

附录 2.7.2 有限差分

del2	拉氏五点差分	gradient	梯度
diff	差分		

附录 2.7.3 向量运算

cross	向量叉积	dot	向量内积

附录 2.7.4 相关

corrcoef	相关系数	subspace	子空间的角度
cov	协方差矩阵		

附录 2.7.5 滤波和卷积

conv	卷积和多项式相乘	filter	一维数字滤波器
conv2	二维卷积	filter2	二维数字滤波器
deconv	解卷和多项式相除		

附录 2.7.6 傅里叶变换

abs	幅值	fft2	二维离散傅里叶变换
angle	相角	ifft	离散傅里叶反变换
cplxpair	复数阵成共轭对排列	ifft2	二维离散傅里叶反变换
fft	快速离散傅里叶变换	nextpow2	最近邻的 2 的幂
fftshift	重排 fft 和 fft2 的输出	unwrap	相位角 360 度线调整

附录 2.8 多项式与插值函数

附录 2.8.1 多项式

conv	多项式相乘	polyval	求多项式的值
deconv	多项式相除	polyvalm	求矩阵多项式的值
poly	由根创建多项式	residue	求部分分式
polyder	多项式微分	roots	求多项式的根
polyfit	多项式拟合		

附录 2.8.2 数据插值

griddata	三维分格点数据	interp1	一维插值
interpft	利用 FFT 方法一维插值	interp2	二维插值

附录 2.8.3 样条插值

spline	三次样条插值	ppval	计算分段多项式

附录 2.9 非线性数值功能函数

fmin	单变量函数最小值	ode23p	解方程并画出解的曲线
fmins	多变量函数最小值	ode45	高阶法解微分方程
fplot	画函数曲线图	quad	低阶法数值积分
fzero	单变量函数的零点	quad8	高阶法数值积分
ode23	低阶法解微分方程	ezplot	绘出方程图形解
dblquad	求二重数值积分		

附录 2.10 二维图形函数

附录 2.10.1 基本平面图形

fill	平面多边形填色	semilogx	X 轴半对数刻度绘图
loglog	双对数刻度曲线	semilogy	Y 轴半对数刻度绘图
plot	直角坐标下线性刻度绘图		

附录 2.10.2 特殊平面图形

bar	直方图	hist	统计频数直方图
compass	从原点出发的复数向量图	polar	极坐标曲线图
comet	彗星状轨迹图	rose	统计频数扇形图
errorbar	误差棒棒图	stairs	阶梯形曲线图
feather	从 X 轴出发的复数向量图	stem	火柴杆图
fplot	函数曲线图		

附录 2.10.3 二维图形注释

grid	画坐标网格线	title	图形标题
gtext	用鼠标在图上标注文字	xlabel	X 轴名标注
legend	图例说明	ylabel	Y 轴名标注
text	在图上标注文字		

附录 2.11 三维图形函数

附录 2.11.1 线、面填色命令

comet3	三维彗星动态轨迹线图	plot3	三维直角坐标曲线图
fill3	三维曲面多边形填色		

附录 2.11.2 三维数据的等高线和其他二维表现

clabel	给等高线加标注	contour3	三维等高线图
contour	等高线图	pcolor	用颜色反映数据的伪色图
contourc	等高线计算	quiver	矢量场图

附录 2.11.3 曲面与网线图

mesh	三维网线图	surfc	带等高线的三维表面图
meshc	带等值线的三维网线图	surfl	带光照的三维表面渲染图
meshz	带零基准面的三维网线图	waterfall	瀑布线曲面图
surf	三维表面图		

附录 2.11.4 切片视图

slice	切片图

附录 2.11.5 图的表现

axis	轴	shading	设置图形渲染方式
caxis	色轴	view	设定 3-D 图形视点
colormap	设置色图	viewmtx	视点转换矩阵
hidden	消隐		

附录 2.11.6 三维图的注释

grid	画坐标网格线	xlabel	X 轴名标注
gtext	用鼠标在图上标注文字	ylabel	Y 轴名标注
text	在图上标注文字	zlabel	Z 轴名标注
title	图形标题		

附录 2.11.7 其他三维图形对象

cylinder	圆柱面	sphere	球面

附录 2.12 通用图形函数

附录 2.12.1 图形窗的产生和控制

clf	消除当前图	figure	打开或创建图形窗口
close	关闭图形	gcf	获得当前图形窗口的句柄

附录 2.12.2 轴的产生和控制

axes	用句柄方法创建轴	gca	获得当前轴的句柄
axis	轴的控制	hold	图形的保持
caxis	控制色轴的刻度	subplot	创建子图
cla	清除当前轴及对象		

附录 2.12.3 图形对象的句柄

axes	创建轴	surface	创建面

figure	创建图形窗口	text	创建文本
image	创建图像	uicontrol	用户界面控制
line	创建线	uimenu	用户菜单控制
patch	创建块		

附录2.12.4 图形句柄操作

delete	删除对象及文件	get	获得对象特性
drawnow	屏幕刷新	newplot	下一个新图属性
findobj	用规定的属性找寻对象	reset	重设对象属性
gco	获得当前对象的句柄	set	建立对象属性
capture	捕捉当前屏的图形	print	打印图形或把图存入文件

附录2.12.5 打印和存储

orient	设置走纸方向	printopt	打印机设置

附录2.12.6 影片与动画制作

getframe	获得影片动画图像的帧	moviein	影片动画内存初始化
movie	播放影片动画		

附录2.12.7 其他图形操作命令

ginput	由鼠标点取图形中坐标	uiputfile	打开文件存储对话框
graymon	设置缺省图形窗口为单色显示屏	uigetfile	打开文件查询对话框
ishold	若图形处保持状态则为真	whitebg	设置窗口背景为白色
rotate	在给定方向上旋转对象	zoom	二维图形的缩放
terminal	设置图形终端类型		

附录2.13 色彩控制和光照模式函数

附录2.13.1 色彩控制

caxis	色轴刻度	shading	颜色渲染模式
colormap	设置色图		

附录2.13.2　色图

bone	蓝色调灰度图	hot	黑-红-黄-白交错色图
cool	青和品红浓淡色图	hsv	饱和色彩图
copper	线性变化纯铜色调图	jet	变异HSV 色图
flag	红-白-蓝-黑交错色图	pink	淡粉红色图
gray	线性灰度	prism	光谱色图

附录2.13.3　色图相关函数

brighten	控制色彩的明暗	rgbplot	取色曲线图
colorbar	色彩条状图		
contrast	提高图像对比度的灰色图	spinmap	颜色周期性变化控制
hsv2rgb	饱和色彩数据向红、绿、蓝数据转换		
rgb2hsv	红、绿、蓝数据向饱和色彩数据转换		

附录2.13.4　光照模式

diffuse	漫反射表面系数	surfl	带光照的三维表面图
specular	漫反射	surfnorm	表面图的法线

附录2.14　特殊矩阵

compan	伴随矩阵	magic	魔方阵
gallery	小测试矩阵	pascal	Pascal 矩阵
hadamard	Hadamard 矩阵	rosser	对称特征值实验问题
hankel	Hankel 矩阵	toeplitz	Toeplitz 矩阵
hilb	Hilbert 矩阵	vander	Vandermonde 矩阵
invhilb	逆Hilbert 矩阵	wilkinson	Wilkinson 的特征值实验矩阵
kron	张量积		

附录2.15　语言结构和调试命令

附录2.15.1　编程语言

eval	字符串宏命令	lasterr	最后一个错误信息

feval	函数宏命令	script	MATLAB 命令文件
function	函数文件头	nargchk	输入变量个数检查
global	定义全局变量		

附录 2.15.2 控制语句

break	终止最内循环	for	循环执行语句
else	同 IF 一起使用	if	条件语句
elseif	同 IF 一起使用	return	返回
end	结束语句	while	重复执行语句
error	显示错误信息		

附录 2.15.3 交互式输入

input	提示键盘输入	uimenu	创建用户界面菜单
keyboard	激活键盘作为命令文件	uicontrol	创建用户界面控制
menu	创建菜单	yesinput	智能键盘输入
pause	暂停		

附录 2.15.4 调试命令

dbclear	清除断点	dbstep	单步执行
dbcont	恢复执行	dbstop	设置断点
dbdown	改变内存的前后关系	dbtype	列出带行号的 M 文件
dbquit	结束调试模式	dbup	改变内存的前后关系
dbstack	列出调用堆栈	mexdebug	MEX 文件调试
dbstatus	列出所有的断点		

附录 2.16 底层文件输入输出函数

附录 2.16.1 文件的开启和关闭

fclose	关闭文件	fopen	打开文件

附录 2.16.2 无格式输入输出

fread	从文件中读二进制数据	fwrite	把二进制数据写入文件

附录2.16.3 有格式输入输出

fgetl	按行从文件读取数据并清除文件指针
fprintf	把格式化数据写入文件
fgets	按行从文件读取数据并保留文件指针
fscanf	从文件中读格式化数据

附录2.16.4 文件定位

ferror	查询文件输入/输出错误状态	fseek	设置文件指针
feof	测试文件是否结束	ftell	得到文件指针
frewind	反绕文件		

附录2.16.5 串的转换

sprintf	写格式数据到串
sscanf	在格式控制下读串

附录2.17 稀疏矩阵函数

附录2.17.1 基本稀疏矩阵

spdiags	从对角阵形成稀疏阵	sprandn	稀疏随机阵
speye	稀疏单位阵	sprandsym	稀疏对称随机库

附录2.17.2 全阵与稀疏阵的转换

find	寻找非零元素下标
sparse	用非零元素和下标创建稀疏阵
full	把稀疏阵转换成全元素阵
spconvert	把稀疏阵转换到指定格式

附录2.17.3 稀疏矩阵非零元的处理

issparse	如果是稀疏阵则为真	spalloc	为非零元素分配存储器
nnz	非零元素个数	spfun	非零元素进行函数计算
nonzeros	非零元素	spones	用1代替非零元素
nzmax	为非零元素分配的存储器数		

附录 2.17.4 稀疏矩阵的可视化

spy	稀疏矩阵的图形表示
gplot	以图论方式作图

附录 2.17.5 排序算法

colmmd	列最小度排序	randperm	随机置换向量
colperm	基于非零算法列排序	symmmd	对称最小度排序
dmperm	Dulmage Mendelsohn 分解		
symrcm	反向 Cuthill-McKee 排序		

附录 2.17.6 范数、条件数和秩

condest	估计范 1 条件数	sprank	结构秩
normest	估计 2 范数		

附录 2.17.7 树的操作

etree	删除树	treelayout	树的展开
etreeplot	画出消除的树	treeplot	树的绘制

附录 2.17.8 关于稀疏矩阵的其他命令

spaugment	形成最小二乘算法系统	spparms	对稀疏矩阵程序设置参数
symbfact	符号因子分析		

附录 2.18 声音处理函数

saxis	声音轴刻度	sound	把向量转换成声音

附录 2.19 动态数据交换函数

ddeadv	设置咨询链接	ddereq	从应用程序提取数据
ddeexec	送一个要执行的串	ddeterm	终止
ddeinit	开始 DDE 对话	DDEddeunadv	释放咨询链接
ddepoke	给应用程序发送数据		

附录 2.20 主启动文件

matlabrc 主启动文件
printopt 设置打印选择

附录 2.21 常微分方程

附录 2.21.1 常微分方程求解器

ode45 用较高阶法解非刚性微分方程
ode23 用低阶法解非刚性微分方程
ode113 用变阶法解非刚性微分方程
ode23t 用梯形法解刚性微分方程
ode15s 用变阶法解刚性微分方程
ode23s 用低阶法解刚性微分方程
ode23tb 用低阶法解刚性微分方程
odefile ODE 文件句法

附录 2.21.2 ODE 选项操作

odeset 创建/改变 ODE 选项结构
odeget 获得 ODE 选项参数

附录 2.21.3 ODE 输出函数

odeplot 时间序列 ODE 输出函数
odephas2 ODE 2 维相平面输出函数
odephas3 ODE 3 维相平面输出函数
odeprint ODE 打印命令窗口

附录 2.22 偏微分方程

附录 2.22.1 PDE 算法

adaptmesh 生成自适应网格和 PDE 解
assema 组装面积积分贡献

assemb	组装边界条件贡献	assempde	组装一个 PD
hyperbolic	解双曲问题	parabolic	解抛物问题
pdeeig	解特征值 PDE 问题	pdenonlin	解非线性 PDE 问题
poisolv	矩形网格上 Poisson 方程的快速解		

附录 2.22.2 用户界面算法和功能

pdecirc	画圆	pdeellip	画椭圆
pdepoly	画多边形	pderect	画矩形
pdetool	PDE Toolbox 的用户界面 (GUI)		
pdemdlcv	将 Matlab 4.2c 的 M-文件转为 Matlab 5.g 格式		

附录 2.22.3 几何算法

csgchk	检查几何描述矩阵的有效性
csgdel	删除最小区域之间的边界
decsg	分解几何结构成最小区域
initmesh	建立初试三角网格
jigglemesh	优化三角网格
pdearcl	弧长和参数表达之间内插值
poimesh	矩形网格上建立规则网格
refinemesh	加密三角网格
wbound	写边界条件的详细数据文件
wgeom	写几何区域的详细数据文件

附录 2.22.4 绘图函数

pdecont	绘制轮廓图的速记命令
pdegplot	绘 PDE 几何图
pdemesh	绘 PDE 三角网格图
pdeplot	PDE Toolbox 的一般绘图函数
pdesurf	绘制曲面图的速记命令

附录 2.22.5 功能算法

dst	离散正弦变换
idst	离散正弦逆变换
pdeadgsc	用相对误差准则选择低劣三角形

pdeadworst	用最低劣值选择三角形
pdecgrad	计算PDE解的通量
pdeent	与已知三角形(集)相邻三角形的索引
pdegrad	计算PDE解的梯度
pdeintrp	插函数值于三角中心处
pdejmps	为适应变化而进行误差估计
pdeprtni	于网格节点处插函数值
pdesde	与子区域集相邻的边界索引
pdesdp	子区域内(节)点的索引
pdesdt	子区域内三角形的索引
pdesmech	计算结构力学张量函数
pdetrg	三角几何数据
pdetriq	测量三角网格质量
poiasma	Poisson方程边界点矩阵贡献
poicalc	矩形网格上Poisson方程的快速解
poiindex	矩形网格正则次序点的索引
sptarn	稀疏特征问题生成的解
tri2grid	将PDE三角网格转化为矩形网格

附录2.22.6 用户自定义算法

pdebound	边界M-文件
pdegeom	几何M-文件

附录2.23 优 化

附录2.23.1 求非线性函数的极小值

fminbnd	矢量约束非线性函数的极小值
fmincon	多维有约束非线性优化
fminsearch	多维无约束非线性优化
fminunc	多维无约束非线性优化
fseminf	多维约束半无穷优化

附录 2.23.2 多目标函数的非线性极小值

fgoalattain 多目标优化
fminimax 多目标最小最大优化

附录 2.23.3 线性最小二乘求解(矩阵问题)

lsqlin 有线性约束条件的线性最小二乘
lsqnonneg 有非负约束条件的线性最小二乘

附录 2.23.4 非线性最小二乘求解(函数问题)

lsqcurvefit 最小二乘曲线拟合
lsqnonlin 有上下限的非线性的最小二乘

附录 2.23.5 非线性方程求解

fzero 标量非线性函数求零点
fsolve 非线性方程求解

附录 2.23.6 矩阵的最小值问题

linprog 线性规划
quadprog 二次规划

附录 2.23.7 求解参数缺省和选择

optimset 建立或改变
optimget 从 OPTIONS 结构获得参数

附录 2.23.8 求极小值

attgoal 多目标寻优
constr 约束条件极值
curvefit 非线性曲线(数据)拟合
fmin 无约束条件极值(标量情况)
fminu 无约束条件极值(用梯度搜索)
fmins 无约束条件极值(简单搜索)
leastsq 非线性最小二乘极值
minimax 最小最大极值

seminf	半无穷极值
lp	线性规划
qp	二次规划
nnls	非负线性最小二乘极值
conls	有约束条件的线性最小二乘
foptions	参数设定

附录2.23.9 三次插值程序

cubic	插值四点求最大值
cubici1	插值两点与其梯度沿直线求最小值
cubici2	插值三点与一个梯度寻优
cubici3	插值两点与其梯度沿直线求最小值和步长

附录2.23.10 二次插值程序

quad2	插值三点求最大值
quadi	插值三点求最小值

附录2.23.11 半无穷规划

semifun	将半无穷多目标问题转换为约束问题
findmax	在一个数据矢量中插值出最大值
findmax2	在一个数据矩阵中插值出最大值
v2sort	将两个矢量排序然后排除漏掉的元素

附录2.23.12 多目标规划

goalfun	将多目标问题转换为约束问题
goalgra	在寻求目标问题中变换梯度

附录2.24 控制系统

附录2.24.1 建立模型

ss	建立状态空间模型
zpk	建立零点/极点/增益模型
tf	建立传递函数模型

dss	定义带描述符的状态空间模型
filt	定义一个数字滤波器
set	设置/修改 LTI 系统的属性
ltiprops	LTI 属性信息

附录 2.24.2　数据提取

ssdata	求状态空间模型系数矩阵
zpkdata	求系统零点、极点和增益
tfdata	取传递函数分子和分母
dssdata	求 dss 的系数矩阵
get	取 LTI 模型属性值

附录 2.24.3　模型特征

isempty	测试 LTI 模型是否为空
isct	测试是否为连续时间系统
isdt	测试是否为离散时间系统
isproper	测试 LTI 模型是否合法
issiso	测试是否为单入单出系统
isa	测试 LTI 模型是否为给定类型

附录 2.24.4　转换

ss	转换为状态空间模型
zpk	转换为零点/极点/增益模型
tf	转换为传递函数模型
c2d	连续型转换为离散型
d2c	离散型转换为连续型
d2d	重新取离散系统或加输入延时

附录 2.24.5　重载运算符

+ and	LTI 系统并联
*	LTI 系统串联
\	左除 sys1 的逆乘以 sys2
/	右除 sys1 乘以 sys2 的逆
'	矩阵转置

.'	非共轭转置
[..]	取LTI系统的行/列向量
inv	系统矩阵求逆

附录 2.24.6　模型动力学

pole, eig	求系统极点
tzero	求系统传输零点
pzmap	零极点图
dcgain	求低频增益
norm	模型正则化
covar	白噪声协方差响应
damp	系统的自然频率及阻尼
esort	返回按实部排序连续系统特征值
dsort	返回按大小排序的离散系统极点
pade	求延时的n阶Pade近似

附录 2.24.7　状态空间模型

rss, drss	建立一个稳定随机状态空间模型
ss2ss	相似变换
canon	求系统标准型
ctrb, obsv	求能控矩阵及能观测矩阵
gram	求格拉姆矩阵
ssbal	求变换阵
balreal	LTI系统的均衡实现
modred	模型降阶
minreal	最小实现
augstate	增广状态空间模型

附录 2.24.8　时域响应

step	阶跃响应
impulse	脉冲响应
initial	给定初始状态的链

附录 2.24.9 连续系统响应

lsim	任意输入的响应
ltiview	响应分析 GUI
gensig	产生 lsim 的输入信号
stepfun	步长函数

附录 2.24.10 频域响应

bode	伯德图
sigma	奇异值 bode 图
nyquist	奈魁斯特图
nichols	尼柯尔斯图
evalfr	求单个给定频率的频率响应
freqresp	求给定频率的频率响应
margin	求增益裕度和相角裕度及相应转折频率

附录 2.24.11 系统连接

append	组合 LTI 系统
parallel	系统并联
series	系统串联
feedback	反馈连接
connect	转换系统框图为状态空间表达式

附录 2.24.12 标准设计工具

rlocus	求根轨迹
rlocfind	指定一组根对应的增益
acker	SISO 的极点配置
place	MIMO 的极点配置
estim	由给定增益设计估计器
reg	由给定的状态反馈及估计器增益设计调节器

附录 2.24.13 LQG 设计工具

lqr,dlqr	线性二次型状态调节器
lqry	带输出加权的 LQ 调节器

lqrd	连续系统的离散 LQ 调节器
kalman	Kalman 估计器
kalmd	连续系统的离散型

附录 2.24.14 估计器

lqgreg	由给定的状态反馈增益及 Kalman 估计器设计 LQG 调节器

附录 2.24.15 矩阵方程的解法

lyap	解连续型 Lyapunov 方程
dlyap	解离散型 Lyapunov 方程
care	解连续型 Riccati 方程
dare	解离散型 Riccati 方程

附录 2.25 信 号 处 理

附录 2.25.1 波形发生

chirp	扫频余弦信号发生器
diric	Dirichlet(周期)函数
gauspuls	Gaussian 脉冲发生器
pulstran	脉冲序列发生器
rectpuls	非周期矩形波发生器
sawtooth	锯齿波函数
sinc	正弦或 sin(pi * x)/(pi * x) 函数
square	方波函数
tripuls	非周期三角波发生器

附录 2.25.2 滤波器分析及实现

abs	取模(绝对值)
angle	相角
conv	卷积
fftfilt	叠加算法的 fft 滤波器
filter	直接滤波器
filtfilt	零相位滤波

filtic	确定滤波器的初始条件
freqs	模拟滤波器的频率响应
freqspace	求频域响应的频率间隔
freqz	数字滤波器的频率响应
grpdelay	求群延迟
impz	脉冲响应(离散型)
latcfilt	格形滤波器
unwrap	相位解卷绕
zplane	离散系统零极点图

附录 2.25.3　线性系统的变换

convmtx	矩阵卷积
poly2rc	由多项式系数计算反射系数
rc2poly	由反射系数计算多项式系数
residuez	Z 变换部分分式或留数计算
sos2ss	2 阶级联形式转换为状态空间表达式
sos2tf	2 阶级联形式转换为传递函数
sos2zp	2 阶级联形式转换为零极点形式
ss2sos	状态空间表达式转换为 2 阶级联形式
ss2tf	状态空间表达式转换成传递函数
ss2zp	状态空间表达式转换成零极点形式
tf2latc	传递函数转换成格形滤波器
tf2ss	传递函数转换成状态空间表达式
tf2zp	传递函数转换成零极点形式
zp2sos	零极点形式转换为 2 阶级联形式
zp2ss	零极点形式转换成状态空间表达式
zp2tf	零极点形式转换成传递函数

附录 2.25.4　IIR 数字滤波器的设计

butter	Butterworth 滤波器
cheby1	契比雪夫 I 型滤波器
cheby2	契比雪夫 II 型滤波器
ellip	椭圆滤波器
maxflat	通用 Butterworth lowpass 滤波器

yulewalk　　递归数字滤波器

附录 2.25.5　IIR 滤波器的阶

buttord　　求 Butterworth 滤波器的阶次
cheb1ord　　求契比雪夫Ⅰ型滤波器的阶次
cheb2ord　　求契比雪夫Ⅱ型滤波器的阶次
ellipord　　求椭圆滤波器的阶次

附录 2.25.6　FIR 滤波器的设计

fir1　　基于窗函数的 FIR 滤波器(标准响应)
fir2　　基于窗函数的 FIR 滤波器(任意响应)
fircls　　约束最小二乘 FIR 滤波器(任意响应)
fircls1　　约束高(低)通最小二乘 FIR 滤波器
firls　　带过渡段的最小二乘 FIR 滤波器(任意响应)
firrcos　　带上升余弦过渡段的 FIR 滤波器
intfilt　　插值 FIR 滤波器
kaiserord　　估计加 Kaiser 窗的 FIR 滤波器的参数
remez　　Parks-McClellan 最优滤波器
remezord　　求 Parks-McClellan 最优滤波器的阶

附录 2.25.7　变换

czt　　线性调频 z 变换
dct　　离散余弦变换
dftmtx　　离散傅里叶变换矩阵
fft　　快速傅里叶变换
fftshift　　fft 输出重排
hilbert　　Hilbert 变换
idct　　逆离散余弦变换
ifft　　逆快速傅里叶变换

附录 2.25.8　统计信号的处理和光谱分析

cohere　　函数相关性
corrcoef　　相关系数矩阵
cov　　协方差矩阵

csd	互谱密度
pmem	MEM(最大扩散法)谱估计
psd	谱密度估计
tfe	传递函数估计
xcorr	互相关函数
xcov	互协方差函数

附录 2.25.9 窗

bartlett	Bartlett window. Bartlett 窗
blackman	Blackman window. 布莱克曼窗
boxcar	Rectangular window. 矩形窗
chebwin	Chebyshev window. 契比雪夫窗
hamming	Hamming window. 海明窗
hanning	Hanning window. 海宁窗
kaiser	Kaiser window. 凯塞窗
triang	Triangular window. 三角窗

附录 2.25.10 参数模型

invfreqs	从频域响应辨识模拟滤波器
invfreqz	从频域响应辨识离散滤波器
levinson	Levinson-Durbin 递归算法
lpc	线性预测系数
prony	时域 IIR 滤波器设计的 Prony 法
stmcb	用 Steiglitz-McBride 迭代法求 ARMA 模型

附录 2.25.11 特殊操作

cceps	倒谱分析和最小相位重构
decimate	降低序列的采样速率
deconv	反卷积和多项式除法
demod	通讯仿真解调
interp	提高采样速率
medfilt1	一维中值滤波
modulate	通信仿真调制
rceps	实倒谱和最小相位重构

resample	改变采样速率
specgram	声谱图
spline	三次样条插值
vco	压控振荡器

附录 2.25.12 相似低通滤波器原型

besselap	Bessel 滤波器原型
buttap	Butterworth 滤波器原型
cheb1ap	契比雪夫 I 型滤波器原型
cheb2ap	契比雪夫 II 型滤波器原型
ellipap	椭圆滤波器原型

附录 2.25.13 频率变换

lp2bp	低通-带通模拟滤波器转换
lp2bs	低通-带阻模拟滤波器转换
lp2hp	低通-高通模拟滤波器转换
lp2lp	低通-低通模拟滤波器转换

附录 2.25.14 滤波器离散化

bilinear	双线性变换
impinvar	(用脉冲响应不变法)模拟-数字滤波器转换

附录 2.25.15 其他

besself	Bessel 模拟滤波器
conv2	二维卷积
cplxpair	按复共轭对排序向量
detrend	删除线性趋势
fft2	二维快速傅里叶变换
ifft2	二维逆快速傅里叶变换
polystab	稳定多项式
stem	绘制离散数据序列
strips	带状图
xcorr2	二维互相关系数

附录 2.25.16 信号 GUI(图形用户界面)

sptool	信号处理工具界面

附录 2.26 simulink 动态仿真系统

附录 2.26.1 Simulation

sim	执行仿真
sldebug	调试仿真模型
simset	置 SIM Options 结构参数
simget	取 SIM Options 结构参数

附录 2.26.2 线性化和平衡分析

linmod	连续系统线性化函数
linmod2	连续系统线性化函数 2
dlinmod	离散系统线性化函数
trim	求平衡点(平衡分析)

附录 2.26.3 模型结构

close_system	关闭仿真窗口或模块对话框
new_system	建立新模型
open_system	打开模型或模块
save_system	保存模型
add_block	加模块到仿真模型
add_line	在模型中连线
delete_block	删除模块
delete_line	删除连线
find_system	查找模型
replace_block	替换模块
set_param	设置参数
get_param	取仿真参数
bdclose	关闭所有仿真窗口

bdroot	返回模块所在系统的根模型名
gcb	取当前模块名
gcbh	取当前模块句柄
gcs	取当前系统名
addterms	加终结器到未连接端口

附录2.26.4 制作模板

hasmask	检查模块是否制成模板
hasmaskdlg	检查模板是否有对话框
hasmaskicon	检查模板是否有图标
iconedit	绘制模块图标
maskpopups	修改模板弹出菜单
movemask	从模板中删除模块

附录2.26.5 模块库

libinfo	取库信息

附录2.27 演示函数

附录2.27.1 引导

expo, demo	启动MATLAB演示屏幕
expomap	打开MATLAB演示主菜单图

附录2.27.2 矩阵

airfoil	airfoil 有限元网格与稀疏阵	inverter	演示矩阵的逆
buckydem	bucky 图与矩阵的关系	matmanip	矩阵运算导论
delsqdemo	不同域内的五点有限差分	sepdemo	有限元网格图
intro	MATLAB 入门	sparsity	稀疏排序的演示

附录2.27.3 数值计算

bench	MATLAB 基准测试程序	odedemo	常微分方程
census	估计美国2000年的人口	quaddemo	求面积(定积分)
eigmovie	求解特征值的动态演示	quake	地震波形

e2pi	e^pi 和 pi^e 哪个大?	rrefmovie	化简行梯形形式的计算
fftdemo	快速离散傅里叶变换的应用	sunspots	答案是 11.08 的问题
spline2d	样条曲线和点的获取演示	fplotdemo	函数绘图
fitdemo	用单纯形算法的非线性拟合	zerodemo	用 fzero 函数寻找零点
funfuns	演示功能函数运算	deedemo1	Van der Pol 方程
deedemo2	Lorenz Atractor	deedemo3	质量和弹簧
deedemo4	质量和弹簧系统动画		

附录 2.27.4 符号数学工具箱

xpcalc	微积分运算	xpgiv	Givens 变换

附录 2.27.5 样条工具箱

spapidm2	样条插值	spldems	样条插值的命令行演示

附录 2.27.6 统计工具箱

statdems	统计工具箱的命令行演示
ppolyt1	对有噪声数据的交互式多项式拟合

附录 2.27.7 图形显示

colormenu	选择色图	lorenz	Lorenz 吸引子
cplxdemo	复变函数图像	membran	创建 Math Works 标记
earthmap	地球的拓扑图	peaks	简单的双变量函数算例
fourier	傅里叶级数展开	penny	对硬币的不同观测
grafcplx	演示复数函数曲线	sqdemo	二次曲面的 UI 控制
graf2d	演示二维曲线	vibes	L 型薄膜振动
graf2d2	演示三维曲线	xpklein	Klein 瓶
imagedemo	图像演示	xpsound	演示声音特性
xpimage	图像处理性能的演示		

附录 2.27.8 图形对象底层操作语言

graf3d	演示表面图的柄图	hndlgraf	演示线图的柄图
hndlaxis	演示轴的柄图	xplang	MATLAB 语言的导论

附录 2.27.9 偏微分方程

pdedemo1	单位圆盘上 Poisson 方程的精确解
pdedemo2	解 Helmholtz 方程和研究反射波
pdedemo3	解最小表面问题
pdedemo4	用于区域分裂解 PDE 问题
pdedemo5	解抛物型方程(热传导方程)
pdedemo6	解抛双曲型方程(波动方程)
pdedemo7	点源自适应问题
pdedemo8	矩形网格上解 Poisson 方程

附录 2.27.10 优化工具箱

bandem	香蕉函数的优化
optdems	优化的命令行演示

附录 2.27.11 控制系统工具箱

ctrldems	控制系统命令行的演示
dskdemo	设置磁盘读/写头控制器
ctrldemo	控制系统工具箱的演示
jetdemo	经典喷气式运输机
diskdemo	硬盘控制器的数字设计
milldemo	轧钢机的 SISO 和 MIMO LQG 控制
kalmdemo	Kalman 滤波器的设计与模拟

附录 2.27.12 鲁棒控制工具箱

accdm2	无摩擦弹簧连接小车系统演示
rctdems	鲁棒控制命令行的演示

附录 2.27.13 信号处理工具箱

filtdem	信号滤波器演示	sigdemo1	信号的离散傅里叶变换
filtdem2	演示滤波器设计	sigdemo2	信号的连续傅里叶变换
filtdemo	滤波器设计	moddemo	调制/解调
sosdemo	二阶分式级联滤波		

附录 2.27.14 简单系统仿真

bounce	弹跳球的 SIMULINK 系统演示
simintro	对 SIMULINK 的入门介绍
libintro	对 SIMULINK 模块库的入门介绍
simppend	SIMULINK 系统单摆模拟
onecart	一辆车的 SIMULINK 仿真演示
vdp	Van der Pol 方程的 SIMULINK 系统演示

附录 2.27.15 复杂系统仿真

dblcart	两辆车的 SIMULINK 仿真演示
f14	F-14 的控制演示
dblcart1	欠阻尼双车系统演示
penddemo	倒摆系统演示
dbblend1	演示双摆系统
thermo	温室控制系统的演示
ldblpend2	演示双摆系统 2

附录 2.27.16 系统辨识工具箱

iddems	系统辨识命令行的演示	sysiddm	辨识毛发干燥器的系统特性

附录 2.27.17 神经网络工具箱

bckprp12	BP 网	neural	特征识别
bckprp62	带动量的 BP 网		

附录 2.27.18 μ-分析和综合工具箱

mudems	设置 μ 分析和综合命令行的演示
xpmu	描述 μ 分析和综合过程

附录 2.27.19 杂项

crulspin	麻花圈旋转	travel	旅行推销员最佳路径问题
logospin	MathWorks 标记动画	truss	支架梁受载动画
makevase	创建镜像图及回转体	wrldtrv	绕球飞行最大环
spinner	彩色线段空间旋转	xpquad	超二次曲面

附录2.27.20 与软件商的联络信息

agents	国际区域联络信息的分布
contact2	怎样用电子信箱与MathWorks联络
contact1	怎样与MathWorks联络
contact3	怎样与MathWorks国际代理联络

参考文献

[1] 张志涌等．掌握和精通MATLAB．北京：北京航空航天大学出版社，1997.
[2] 陆君安等．偏微分方程的MATLAB解法．武汉：武汉大学出版社，2000.
[3] 施阳等．MATLAB语言工具箱—ToolBox实用指南．西安：西北工业大学出版社，1998.
[4] 魏克新等．MATLAB语言与自动控制系统设计．北京：机械工业出版社，1997.
[5] 高均斌．MATLAB 5.0 语言与程序设计．华中理工大学出版社，1998.
[6] 李涛等．MATLAB工具箱应用指南——应用数学篇．2000.
[7] 陈立周等．机械优化设计．上海：上海科学技术出版社，1982.12.
[8] 郭应龙主编．机械动力学．北京：水利电利出版社，1994.6.
[9] (日)户川隼人著．殷荫龙，陈学源译．振动分析的有限元．北京：地震出版社，1985.5.